FEATURE DISTRIBUTION LEARNING FOR COVARIATE SHIFT ADAPTATION USING SPARSE FILTERING

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2017

By
Fabio Massimo Zennaro
School of Computer Science

Contents

Word Count: 64294

List of Tables

List of Figures

Abbreviations

AE Auto-Encoders.

CD Cross-domain Difference.

CSA Covariate Shift Adaptation.

DAE Denoising Auto-Encoders.

DDL Data Distribution Learning.

EMODB Berlin Emotional Data Set.

ESR Emotional Speech Recognition.

FDL Feature Distribution Learning.

GMM Gaussian Mixture Models.

i.i.d. Independent and Identically Distributed.

ICA Independent Component Analysis.

IW Importance Weighting.

IWLR Importance-Weighting Logistic Regression.

IWLS Importance-Weighting Least Squares.

IWSVM Importance-Weighting Support Vector Machine.

KDE Kernel Density Estimation.

KLIEP Kullback-Leibler Importance Estimation Procedure.

KMM Kernel Mean Matching.

KNN K-Nearest Neighbours.

KS Kolmogorov-Smirnov.

L-BFGS Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm.

LSIF Least-Squares Importance Fitting.

LSPC Least-Square Probabilistic Classifier.

MFCC Mel-Frequency Cepstrum Coefficient.

MMD Maximum Mean Discrepancy.

MTL Multi-Task Learning.

OLS Ordinary Least Squared.

PCA Principal Component Analysis.

PD Percentage Drop.

pdf Probability Density Function.

pmf Probability Mass Function.

PPD Performance Percentage Difference.

PSF Periodic Sparse Filtering.

RBM Restricted Boltzmann Machines.

ReLU Rectified Linear Unit.

RL Representation Learning.

SF Sparse Filtering.

SSA Sub-Space Alignment.

SVM Support Vector Machine.

UAR Unweighted Average Recall.

Notation

Generic Notation

$\propto$	Proportional
$\equiv$	Equivalent
$\circ$	Function composition
$\{\cdot\}$	Set
$\cup$	Set union
$\cap$	Set intersection
$\setminus$	Set subtraction
$\times$	Set Cartesian product
$[\cdot, \cdot]$	Closed interval
$(\cdot, \cdot)$	Open interval
$a, b, c...$	Generic scalars
$f(\cdot), g(\cdot), h(\cdot)...$	Generic functions
$f : \mathbb{R} \to \mathbb{R}$	Generic definition of function
$\mathbf{a}, \mathbf{b}, \mathbf{c}...$	Generic vectors
a_i	i^{th} element of the the vector a
$\mathbf{A}, \mathbf{B}, \mathbf{C}...$	Generic matrices
$\mathbf{A}_{[i,j]}$	$(i,j)^{th}$ sub-matrix of $\mathbf{A}$
$\mathbf{a}_i$	i^{th} column of the the matrix $\mathbf{A}$ (sample)
$\mathbf{a}_{\cdot,j}$	j^{th} row of the the matrix $\mathbf{A}$ (feature)
$a_{i,j}$	$(i,j)^{th}$ element of the the matrix $\mathbf{A}$
$[a_{i,j}]$	Matrix containing the elements $a_{i,j}$
$A, B, C...$	Generic random variables
$p(A), p(B), p(C)...$	Generic probability distribution functions
$\mathtt{A}, \mathtt{B}, \mathtt{C}...$	Generic algorithms
$\mathbb{I}, \mathbb{N}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$	Integer, natural, rational, real, complex domain
$\mathbb{R}_{\geq 0}, \mathbb{R}_{>0}$	Non-negative, strictly positive domain
$\mathfrak{F}$	Space of functions
$\mathfrak{P}$	Space of pdfs

Functions and Matrix Notation

$\lvert\cdot\rvert$	Absolute-value function
$\ell_0(\cdot), \ell_1(\cdot), \ell_2(\cdot), ..., \ell_p(\cdot)$	ℓ_0-, ℓ_1-, ℓ_2-, ℓ_p-norms
$\ell_{p,\epsilon}(\cdot)$	ℓ_p-norm with ϵ threshold
$\mathbf{1}_c(\cdot)$	Indicator function with condition c
$\text{ReLU}(\cdot)$	Rectified linear unit function
$\sigma(\cdot)$	Sigmoid function
$D_A[x, y]$	Distance A between x and y
$\mathbf{I}$	Identity matrix
$\odot$	Element-wise multiplication
$.^\top$	Transposition operator
$\mathbf{e}_k$	k^{th} orthonormal basis
$\min_D P, \max_D P$	Minimization, maximization of P over domain D
$\operatorname{argmin}_D P, \operatorname{argmax}_D P$	Arg-minimization, arg-maximization of P over domain D
$\mathcal{L}$	Loss function
$\mathcal{P}$	Performance index
$\mathcal{O}(\cdot)$	Big-O notation for computational complexity
$\mathcal{M}(n)$	Time complexity for multiplication

Probability and Statistics Notation

$\sim$	Distributes as...
$\approx$	Approximates...
$\hat{=}$	Estimated as...
$p(X)$	Probability distribution function. Shorthand for $p(X = x)$ or $p_X(x)$
$p(X, Y)$	Joint distribution function
$p(X\|Y)$	Conditional distribution function
$F_X(\cdot)$	Cumulative distribution function
$\hat{\cdot}$	Empirical estimation
$M_i[X]$	i^{th} moment of the random variable X
$E[X]$	Expected value of the random variable X
$Var[X]$	Variance of the random variable X
$H_A[X]$	A-entropy of the random variable X
$H_A[X\|Y]$	A-conditional entropy of the random variable X given Y
$H_A[X;Y]$	A-cross-entropy of the random variables X and Y
$MI[X;Y]$	Mutual information of the random variables X and Y
$D_A[p \parallel q]$	A-divergence between the pdfs p and q
$D_A[p, q]$	A-distance between the pdfs p and q
$\mathcal{N}(\mu, \mathbf{\Sigma})$	Gaussian pdf with mean μ and covariance $\mathbf{\Sigma}$
$\mathcal{U}(a, b)$	Uniform pdf within the interval $[a, b]$

Data Notation

$\mathbf{X}$	Original data ($\mathbf{X} \in \mathbb{R}^{M \times N}$)
$\mathbf{Z}$	Learned representations ($\mathbf{Z} \in \mathbb{R}^{L \times N}$)
$\mathbf{Y}$	Labels ($\mathbf{Y} \in \mathbb{R}^{N}$)
X	Set of data
$\mathcal{X}$	Domain of data
$\mathbf{X^{tr}}, \mathbf{X^{tst}}, \mathbf{X^{val}}, \mathbf{X^{tgt}}, \mathbf{X^{src}}$	Training, test, validation, target, source data
$\mathbf{X^{sup}}, \mathbf{X^{unsup}}$	Supervised, unsupervised data
$\mathbf{X}^*$	Data from true (noiseless) distribution
X	Random variable for the original data
$p(X)$	Pdf for the original data
$X^{tr}, X^{tst}, X^{val}, X^{tgt}, X^{src}$	Random variable for training, test, validation, target, source data
$X_{\cdot,j}$	Random variable for the j^{th} feature
N	Number of samples
M, L	Number of features of original and learned representations
C	Number of labels
$\mathbf{W}$	Weight matrix ($\mathbf{W} \in \mathbb{R}^{L \times M}$)
Θ	Set of parameters
θ_i	i^{th} parameter
Ξ	Set of hyper-parameters
ξ_j	j^{th} hyper-parameter

Sparse Filtering Notation

$\ell_{2,row}(\cdot), \ell_{2,col}(\cdot)$	ℓ_2-norm along the row and along the columns
$f_{A1}, f_{A2}, f_{A3}, f_{A4}, f_{A5}$	Steps A1, A2, A3, A4, A5 of the SF algorithm
$f_{A1:A4}$	Composition of steps A1 to A4 of the SF algorithm
$\mathbf{F}, \tilde{\mathbf{F}}, \hat{\mathbf{F}}$	Outputs of steps A2, A3 and A4 of the SF algorithm
$\mathbf{c}$	Vector of normalization constants in step A3
$\mathbf{b}$	Bias vector
$\bar{\mathbf{x}}$	Displacement vector
λ_i	Vector of scaling constants
$\mathbf{k}$	Vector of periodic constants
$R^{\mathbf{e}_k}$	Representation filter for the k^{th} orthonormal basis
$R^{\mathbf{e}_k}_{\mathbf{x}_i}$	Representation filter for the orthonormal basis centred on the point $\mathbf{x}_i$

Abstract

Feature Distribution Learning for Covariate Shift Adaptation
Using Sparse Filtering
Fabio Massimo Zennaro
A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy, 2017

This thesis studies a family of unsupervised learning algorithms called *feature distribution learning* and their extension to perform *covariate shift adaptation*. Unsupervised learning is one of the most active areas of research in machine learning, and a central challenge in this field is to develop simple and robust algorithms able to work in real-world scenarios. A traditional assumption of machine learning is the independence and identical distribution of data. Unfortunately, in realistic conditions this assumption is often unmet and the performances of traditional algorithms may be severely compromised. Covariate shift adaptation has then developed as a lively sub-field concerned with designing algorithms that can account for covariate shift, that is for a difference in the distribution of training and test samples.

The first part of this dissertation focuses on the study of a family of unsupervised learning algorithms that has been recently proposed and has shown promise: *feature distribution learning*; in particular, *sparse filtering*, the most representative feature distribution learning algorithm, has commanded interest because of its simplicity and state-of-the-art performance. Despite its success and its frequent adoption, sparse filtering lacks any strong theoretical justification. This research questions how feature distribution learning can be rigorously formalized and how the dynamics of sparse filtering can be explained. These questions are answered by first putting forward a new definition of feature distribution learning based on concepts from information theory and optimization theory; relying on this, a theoretical analysis of sparse filtering is carried out, which is validated on both synthetic and real-world data sets. In the second part, the use of feature distribution learning algorithms to perform *covariate shift adaptation* is considered. Indeed, because of their definition and apparent insensitivity to the problem of modelling data distributions, feature distribution learning algorithms seems particularly fit to deal with covariate shift. This research questions whether and how feature distribution learning may be fruitfully employed to perform covariate shift adaptation. After making explicit the conditions of success for performing covariate shift adaptation, a theoretical analysis of sparse filtering and another novel algorithm, *periodic sparse filtering*, is carried out; this allows for

the determination of the specific conditions under which these algorithms successfully work. Finally, a comparison of these sparse filtering-based algorithms against other traditional algorithms aimed at covariate shift adaptation is offered, showing that the novel algorithm is able to achieve competitive performance.

In conclusion, this thesis provides a new rigorous framework to analyse and design feature distribution learning algorithms; it sheds light on the hidden assumptions behind sparse filtering, offering a clear understanding of its conditions of success; it uncovers the potential and the limitations of sparse filtering-based algorithm in performing covariate shift adaptation. These results are relevant both for researchers interested in furthering the understanding of unsupervised learning algorithms and for practitioners interested in deploying feature distribution learning in an informed way.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to start by thanking my supervisor, Dr Ke Chen, for his constant help and support during the four years of my PhD. I am very grateful for his generous and expert guidance, and for allowing me to pursue interesting research topics and to grow as a researcher. I am extremely thankful both for his continuous direction and for his understanding in difficult times.

My doctoral experience would have not been so rich without the contribution of all the people working at the University of Manchester. My gratitude goes to the staff at the School of Computer Science for their helpfulness; to the academic members of the MLO group for setting up a stimulating environment in which to do research; to Dr Erica Baffelli and Dr Todd Klutz for their enriching classes; and to all the colleagues who shared with me the life of being a PhD student: Ubai, Peizhi, Harits, Qian, Jon, Colin, Henry, Kostas, Nikos, Sarah, Andrew, Joe, Fanlin and Vassilis.

Outside the university, I wish to thank the people I lived with during these years: all the residents of the House of St Gregory and St Macrina, and particularly Dimitrios, for his conversation, and Laurence, for the many hours spent in the library together. Too much space would be required to thank all the friends who blessed me with their friendship and support, but I can not avoid expressing fond gratitude to my Italian friends: Defra, Luca, Ste, Codo, Pisa, Dario and Federico; I am so grateful for how they are always ready to welcome me back among them every time I go back, as if I had never left.

The most special thanks is to my wife, Emily. Not so much for her proofreading, her support and her patience, but mainly for being always there. Her presence and her love have been the light of these years.

Finally, the deepest thanks goes to my uncle and my parents. Especially my parents for their unwavering love: always present whenever I needed, always understanding whatever path I chose. Even when my choices took me away from them. Thank you to my father, who is supporting me every single day. Thank you to my mother, whose love was really unbounded.

The threads in a kind mother's hand
A gown for her son bound to a far off land,
sewn stitch-by-stitch before he leaves,
for fear his return may be delayed.
Such a kindness as a young grass receives
from the warm sun can not be repaid.
— Meng Jiao (trans. Xu Yuanchong)

Chapter 1

Introduction

This chapter introduces the work of the present dissertation with the aim of giving an overview of the motivation and the objectives behind this research.

Section 1.1 situates this dissertation within the larger context of machine learning and unsupervised learning. Section 1.2 presents the specific research questions that drove and justified our research. Section 1.3 examines some of the methodological approaches used in this research and while writing this dissertation. Section 1.4 offers a summary of the main results and contributions of our work. Section 1.5 explains the structure of the ensuing dissertation. Section 1.6 lists the publications that were produced in the course of this research.

1.1 Relevance of the Thesis

This section offers a presentation of the research done in the context of this dissertation, with a particular focus on situating this work within the wider field and highlighting the main directions of this study.

In general terms, this dissertation aims to offer a contribution to the field of machine learning, and, more specifically, to foster the theoretical understanding and the practical development of unsupervised learning algorithms fit to work in real-world scenarios.

Machine learning and unsupervised learning. As a field, machine learning is focused on the problem of developing machines able to make sense of data. In more formal terms, machine learning tackles the problem of making computers able to perform sound inferences from large amount of data. Classically, it is possible to distinguish two main settings for the problem of extracting knowledge from data.

In the first setting, inference is hetero-directed through the provision of human information: a set of data is accompanied by a token of meaningful information, and the machine is expected to learn a rule that generalizes the association between the data and the knowledge provided by the tokens. This is supervised learning.

In the second setting, inference has to be performed in an auto-directed way by a machine: no external information is provided, and the learning machine is expected to extract knowledge

on the basis of statistical properties of the data. This is unsupervised learning.

In recent years, the amount of data available has kept increasing at an ever faster pace, while the human ability to provide machines with meaning has been substantially constant, constrained by time and monetary bounds. As such, the interest in unsupervised learning and in the possibility of developing machines able to learn without human supervision has steadily increased. Unsupervised learning has been recognized, at the same time, to be a necessary tool for dealing with the overflow of data and as a crucial step on the way to the development of truly intelligent systems. Several different aspects of unsupervised learning have been the object of research: from developing theoretical framework to ground the behaviour of already existing algorithms to devising novel and more efficient solutions, from extending the domain of applicability of standard models to the application of learning systems to real-world scenarios.

Feature distribution learning and covariate shift adaptation. This dissertation is interested in the study of a specific family of algorithms within the field of unsupervised learning, that is, feature distribution learning (FDL) algorithms, and in the exploitation of these algorithms to develop robust systems that can perform covariate shift adaptation (CSA).

FDL is a recently-developed approach to unsupervised learning focused on processing data in order to learn new representations optimized for specific types of tasks. What sets FDL algorithms apart from traditional unsupervised learning algorithms is their unusual approach to data processing: instead of explicitly modelling the data available, they just aim to generate new representations with useful properties. This approach allows FDL algorithms to circumvent the hard problem of estimating the distribution that generates the data and allows them to focus instead on solving a potentially simpler problem. FDL algorithms can be very efficient in terms of computational complexity and they can smoothly scale up to large amount of data. These features are extremely attractive from a practical point of view and may lead to a wider adoption of these algorithms in the future. So far, despite the promise shown by FDL algorithms in empirical studies, no thorough theoretical study has been provided to explain and justify these algorithms.

CSA denotes a wide range of algorithms aimed at tackling the very common, but often unacknowledged, problem of covariate shift. Covariate shift is a ubiquitous phenomenon in machine learning, caused by a mismatch between the data on which a machine was trained and the data that the machine needs to process at test or deployment time. Indeed, a model learned on a given set of data can generalize only with respect to instances coming from the same process that generated the data used for learning. If the generating process were to change between training and deployment, the model would no longer be guaranteed to explain the data. This mismatch arises frequently in real-world scenarios, such as when an algorithm is trained in a certain environment and then deployed in a new setting. For instance, a speech recognition model trained on a set of speakers recorded in a noiseless environment using a rigorous experimental protocol may utterly fail to recognize speech collected from new users on the street. CSA algorithms try to solve the problem of covariate shift by adapting the data in a way that a generalization process can be safely extended to data generated by a similar but not identical process. Given the expectation that machine learning models comply

with complex and changing environments, CSA has become a relevant topic of research and an important component of robust algorithms that are designed to be deployed in real-world scenario. Research on the topic of CSA is particularly active and has become object of significant attention.

Joining feature distribution learning and covariate shift adaptation. This work aims to bring together, in a grounded and methodical way, the two strands of research on FDL and CSA. In the first part of our research, we offer an analysis of FDL algorithms, explaining their dynamics, their inner working, their strengths and their limitations. Relying on this understanding, in the second part we investigate the possibility of exploiting these properties to tackle the problem of covariate shift.

Our research revolves mainly around a specific case study, that is the sparse filtering (SF) algorithm. SF remains, to date, the prototypical FDL algorithm and the only widely-acknowledged FDL algorithm. Beyond its uniqueness, SF has gathered attention in the research community because of its sheer success. Its efficacy (proved by its state-of-the-art results) and its efficiency (in terms of simple coding, few modelling hyper-parameters, computational speed) are good enough reasons to justify the study of this algorithm. Indeed, since its introduction, SF has been successfully used in machine learning contests, it has been edited and modified, and it has been integrated in many real-world applications. Nevertheless, SF has been the object of very few theoretical and experimental studies.

Our study of SF will be instrumental and illustrative for the study of FDL in general. In the first part of our research, we provide a deep theoretical analysis and an extensive empirical validation of the SF algorithm. We take, however, special care so that our results are not limited to the particular case of SF. We offer discussion and insights on how our conclusions on SF can be, on a larger scale, applied to the entire class of FDL algorithms. In the second part of our research, we start from our new grounded understanding of SF to explore the possibility of applying FDL to the problem of covariate shift. SF will be the starting point for the development of a novel FDL algorithm able to perform effective CSA, both on synthetic and real-world data.

In the end, our work is meant to provide a deeper and more grounded understanding of FDL algorithms in general, and of the SF algorithm in particular. Our results will offer a strong justification and reliable guidelines for using these algorithms both under standard simplified learning assumptions and under the more realistic assumption of covariate shift. We expect our conclusions to be of use both for the more applied practitioner as well for the theoretically-versed researcher. The first can gather useful insights on the strengths and the limitations of the existing algorithms in order to be able to choose when to use them in an informed way. The second can build upon our conceptual and theoretical framework to analyse FDL algorithms at a deeper level or to develop new algorithms in a grounded way.

Given this presentation of the field, the next section will discuss the research questions that specifically drove our work.

1.2 Research Questions

This section describes the research questions at the foundation of our research. We list these central questions, explaining each one and enumerating the specific and particular sub-questions that they raise.

In the first part of our work, we focus on the analysis of FDL algorithms in general terms. We start asking a high-level research question related to FDL algorithms:

- *How can we rigorously define a FDL algorithm?*
  FDL was introduced as a novel conceptual category meant to embrace unsupervised algorithms that perform learning without caring about the problem of modelling the distribution of the data. This definition, even if intuitively clear, presents some shortcomings when analysed more closely. In particular, it seems to lack the necessary rigour to definitively distinguish algorithms that are supposed to be FDL algorithms from those that are not. Before proceeding in the study of any FDL algorithm, it is reasonable to address the problem of what we actually mean by FDL. This leads us to reflect on the original definition of FDL and ask the following questions: *what does it mean to ignore the problem of learning the data distribution? Is it possible to actually and completely ignore this? If not so, to what degree can it be ignored? How can we express this idea more formally?*

Investigating these questions leads us to offer a more precise definition of FDL algorithms, relying less on intuitive terms and more on the language of information theory and optimization theory.

The following natural step is to consider real and concrete implementations of FDL algorithms. Our attention inevitably focuses on SF, as the most representative and successful instance of a FDL algorithm. SF provides a very interesting case study because, despite its success, no strong justification has been provided to explain its results, beyond, again, intuitive descriptions. We then formulate the following research question:

- *Can we explain the behaviour of the SF algorithm through the lens of our redefined conceptual and theoretical understanding of FDL algorithms?*
  The efficacy of unsupervised representation learning algorithms is normally explained in terms of the quality of the learned representation. However, differently from other algorithms but according to the FDL paradigm, SF seems to make no explicit reference to the objective of learning representations related to the original data. Inevitably, when considering the SF algorithm and its dynamics, some immediate questions are raised: *how does SF provide useful and meaningful representations? How are the representations learned by SF related to the original representations? What is the role and the relevance of each step of the SF algorithm in the overall learning? How could SF be put in relationship to other unsupervised algorithms?*

The conceptual and theoretical understanding of FDL that we develop proves extremely useful in tackling all these questions about SF and its properties. Indeed, it provides a clear direction and a useful strategy for studying SF, allowing us to highlight and explain the actual mechanics of the algorithm.

The success of this analysis invites us to try to extend our conclusions beyond the narrow case of SF. As no other algorithm beyond SF has been unanimously identified as belonging to the FDL family, we wonder if our study may allow us to consider or devise other potential FDL algorithms. We express this objective in the following research question:

- *Building upon our newly-developed understanding of the SF algorithm, can we identify, explain or design alternative FDL algorithms?*
  This question requires us to address at least two different sub-problems; the first is whether existing algorithms in the literature can be interpreted as forms of FDL algorithms; the second is whether we can exploit our understanding of SF to develop new FDL algorithms. These aims automatically translate into more specific questions: *how could the SF algorithm be modified? Within which limits can we develop new FDL SF-like algorithms? What other existing algorithms qualify as FDL?*

Working on these questions allows us to broaden our conclusions from the limited case of SF to other potential FDL algorithms. This study lays foundations for future work aimed at developing novel FDL algorithms which could be designed to tackle specific problems.

All these results are exploited and further developed in the second part of our work, in which we investigate the possibility of using FDL algorithms to perform CSA. Once again we start with a high-level question that informs all our ensuing research:

- *Can FDL be successfully applied to CSA, and, if so, how?*
  As we will explain, there are interesting and compelling reasons to expect FDL algorithms to be useful in performing CSA. However, we want to move beyond a mere intuition and investigate more formally whether and when FDL may be a sensible choice for performing CSA, especially in a classification scenario. More precisely, we set out to answer the following basic questions: *is it possible to perform CSA with no reference to the data, as the FDL paradigm promises? Under which conditions can an unsupervised algorithm in general, and a FDL algorithm specifically, successfully perform CSA?*

Examining these questions leads us to clarify what are the requirements that an unsupervised learning algorithm must satisfy in order to perform CSA. In other words, it allows us to define the conditions of success that we can use to study and evaluate the possibility for FDL algorithms to perform CSA.

In order to exploit the results we have obtained before, we begin by considering whether these generic conditions for CSA can be met by the most representative FDL algorithm, that is SF. We formulate this problem through the following research question:

- *Can we perform CSA via SF?*
  This question requires us to evaluate SF not only in general terms, but on a specific non-ideal working condition defined by covariate shift. To do this, we bring together the results

of our theoretical study of SF and our new understanding of CSA via FDL. Practically, we ask ourselves the following questions: *does SF meet the conditions for CSA? What are the cases in which SF provides good results? How does SF fare against other CSA algorithms? Is SF effective in real-world scenarios?*

This study unveils the possibilities and the limitations of SF for performing CSA. Unfortunately, it turns out that the conditions under which SF is guaranteed to perform CSA are quite strict, and, consequently, few concrete scenarios may be expected to conform.

Faced with this severe limitation, we are then prompted to wonder whether alternative FDL algorithms may guarantee better performance in the presence of covariate shift. We pose then the following research question:

- *Can we find a FDL algorithm that can overcome the limitations of SF?*
  Aware of the restrictions of SF, we consider the possibility of finding or designing an alternative FDL algorithm to perform CSA. Starting from the basic SF algorithm, we devise a new FDL algorithm able to perform CSA under more versatile conditions. To do so, we address the following sub-questions: *in which scenarios do we want the new FDL algorithm to work? Can we validate theoretically its conditions of success as we did for SF? Can we apply this new algorithm to real-world problems?*

Tackling this last question requires us to exploit the knowledge that we developed in both the first and second part of our research. Our study of FDL algorithms provides us now with a reliable and sound understanding of how to develop new FDL algorithms, while our study of CSA provides us with clear and specific conditions that a new FDL algorithm must meet in order to successfully perform CSA. Putting these results together allows us to seamlessly define a new SF-like algorithm able to perform CSA under looser conditions than SF.

In the next section we will review some of the methodological choices that inform our research.

1.3 Methodology

This section makes explicit some of the methodological practices followed in our research.

Our approach to the study of FDL and CSA is based on the use of diverse conceptual, theoretical and experimental tools.

Conceptual analysis. On the conceptual level, we always aim to provide clear definitions of concepts, tools and aims. We put particular emphasis on giving, at the same time, rigorous definitions and meaningful interpretations of our choices and results. Important terms that constitute a conceptual foundation for this work and that are relevant in the following explanations and interpretations are capitalized throughout the background chapter.

Whenever providing a conceptual narrative, we try to follow a consistent approach. In particular, in Chapter 2, we follow a coherent top-down approach in the description of the machine

learning field and in introducing our specific research. In trying to survey a conceptual landscape, we recognize that two different directions may be followed. The first one is a bottom-up approach, in which specific instances of concepts and problems are presented and discussed one by one; these particular concepts may then be aggregated through a synthetic effort into overarching ideas that encompass specific problems. The second approach is a top-down approach, in which general and abstract concepts are considered and examined; the generic notions may then be refined, through an analytic effort, into more concrete cases. While the first approach has the advantage of starting from concrete problems that may be relevant to the community, the second has the advantage of providing a consistent and potentially exhaustive framework within which to organize our ideas. These two approaches are clearly abstractions which can be strictly followed only to a certain degree, as analytical and synthetic movements will inevitably alternate. However, in our presentation we try to adhere, as closely as possible, to a top-down approach.

Extreme care has been put also on expressing and making explicit all the assumptions made in our reasoning. We consider this a very important point, as the validity and the extent of any conclusion we reach is inevitably bound and limited by the assumptions on which it is grounded. Even if often disregarded, assumptions are particularly critical in machine learning, where the statistical validity of conclusions is often strictly dependant on such assumptions. Expressing every assumption is of course a challenging task which requires careful and continuous self-inquiry. Despite our best efforts, we are sure several reasonable assumptions that are easily taken for granted have remained unexpressed. However incomplete, we believe this to be a worthwhile effort and, for this reason, we have given particular relevance to assumptions by highlighting them in the text.

Theoretical analysis. On the theoretical level, we put particular care into the rigour of the mathematical formulations expressing our algorithms. We introduce all the mathematical terms we borrow from different areas of research and provide clear definitions for new terms that we adopt. We define properties in propositions and theorems, and we prove them rigorously.

Experimental analysis. On the experimental level, experiments are carefully designed to validate our theoretical propositions. Simulations are run both on synthetic and real-world data sets. The first are designed to demonstrate and confirm our statements in an immediate way; we often use simple data sets devised to be easily visualizable and illustrative. The second ones are comprised of real-world data sets available to the machine learning community; in this case we study data sets that were previously studied in the literature and data sets that comply with the assumptions of our algorithms. For all the simulations, both with synthetic and real-world data, we provide precise experimental details, in order to guarantee the full reproducibility of our results. In the main text or in appendices, we provide links to our source code and, when possible, links to other open source code and data sets used in the simulations.

Having summarized the objectives and the methodology, we will review in the next section the contributions provided by our work.

1.4 Contributions of the Thesis

This section details the results and the contributions provided by our research.

Concerning the study of FDL algorithms our main contributions may be summarized as follows:

- We review the definition of FDL algorithms and propose an alternative and more rigorous definition using the language of information theory and optimization theory. We propose to explain the behaviour of FDL algorithms in terms of objectives and constraints aimed at maximizing information and preserving the structure of the data. On a practical level, this new understanding of FDL algorithms provides useful directions and insights for analysing existing algorithms, devising new algorithms, or simply interpreting other algorithms through the lens of FDL.

- Following the conceptual framework described above, we carry out a formal analysis of the SF algorithm, which allows us to understand and explain how SF generates new representations. We show how SF preserves information through the proxy of sparsity, and how relevant structure is retained when the data may be explained by a metric of cosine neighbourhoodness. In this way, we provide a neat theoretical justification of the so-far-unexplained success of SF, and we discover under which conditions SF may be expected to be useful and effective. Furthermore, we validate all our statements through simulations on synthetic and real-world data.

- We extend our results and insights from the case study of SF to other potential FDL algorithms. In particular, we show the possibility of developing new SF-like algorithms and, at the same time, we point out the limitations of some potential alternative algorithms. Reasoning in terms of structure preservation, we show that intuitive modifications of the original SF algorithm (such as the implementation of SF-like algorithms using classical non-linearities from the neural network literature) are bound, by a theoretical argument, to under-perform. Our FDL framework also allows us to interpret some existing algorithms (such as random projection algorithms) in light of structure preservation; this may be extended to other algorithms in the future and may aid developing a better understanding of the dynamics of representation learning.

Concerning the study of CSA, we show how our deep understanding of FDL algorithms may be exploited to develop algorithms able to tackle real and challenging problems such as covariate shift. In this regard, our main contributions can be summarized as follows:

- We formally define the conditions for successful CSA for FDL algorithms. This includes not only the trivial compensation for the difference in the distribution of the data, but also the critical requirement of preservation of the conditional distribution of the labels. Clarifying these requirements is crucial for a rigorous and precise analysis of the potential for FDL algorithm to perform CSA.

- We show that SF has a limited potential for CSA. On one hand, we prove that SF can implicitly reduce the distance between the marginal distributions of the data; on the

other hand, however, we show that its ability to preserve the conditional distribution of the labels is intimately connected to its property of structure preservation. Relying on the results obtained in our study of SF, we are then able to show that a conditional distribution with a radial structure is required for SF to perform successful CSA.

- We overcome the limitations of standard SF by designing a new algorithm called PSF. Through a formal analysis, we show that PSF can reduce the distance between the distribution of the data in the same way SF does; moreover, we show that it can preserve the conditional distribution of the labels whenever the data exhibit a periodic behaviour. These more flexible dynamics makes PSF particularly suitable for dealing with data sets that are affected by a form of covariate shift caused by their dependence on the sampled users. These results are confirmed on synthetic and real-world data in experiments that allow us to point out both the limitations and strengths of both SF and PSF alongside other classical CSA algorithms.

Overall, our research makes contributions on a conceptual level (offering a clearer and more rigorous understanding of FDL algorithms and the requirements for their application to CSA), on a theoretical level (uncovering the properties and the dynamics of SF and discussing ways in which they could be exploited to develop new algorithms) and on a practical level (showing the advantages and the limitations of FDL algorithms, such as SF, sigmoid SF, or PSF). These results may be of use both for researchers, who could build upon our results to pursue an even better and more formal understanding of FDL and SF, and to practitioners, who may be required to decide whether the adoption of FDL algorithms like SF would be a good or a bad choice in their domain.

In the next sections, we will outline the structure of this dissertation and list the publications that followed from our research.

1.5 Outline of the Thesis

The rest of the dissertation is organized as follows.

Chapter 2 provides a solid background to the work done in this dissertation by offering a bird's eye view of the field of machine learning and by locating and introducing the specific topics of this thesis. It discusses the problem of learning in generic terms first, and then in formal terms. Next, it introduces the two main topics of our research: FDL algorithms and learning under covariate shift. Finally, it outlines the open problems presented in these fields and, in particular, the challenges and the opportunities arising from the meeting of the two topics of FDL and CSA.

Chapter 3 and 4 contain the main contributions of this dissertation.

Chapter 3 is focused on the study of the promising family of FDL algorithms. First, it proposes a more sound conceptual definition of data distribution learning and FDL algorithms. Based on this understanding, a new and deep theoretical and empirical analysis of the most important representative FDL algorithm, that is, SF, is carried out. This study allows the

reader to gain a strong insight about SF, to understand its strengths and limitations, and to align and compare it with other machine learning algorithms. Finally, always relying on this conceptual framework, a discussion on how alternative SF-like algorithms may be designed and how already existing algorithms may be understood as instances of the FDL family is presented.

Chapter 4 studies how FDL algorithms may be applied to the problem of covariate shift. Once again, this chapter starts with the definition of a conceptual framework specifying the conditions of success for performing CSA using FDL algorithms. In light of this understanding, a rigorous analysis on the ability of SF to perform CSA is performed. Faced with the limitations of this algorithm, a novel SF-like algorithm called PSF is introduced, and, as in the case of SF, a rigorous theoretical analysis of its ability to perform CSA is conducted. Finally, a validation of the theoretical results and a comparison against other well-known CSA algorithms from the machine learning literature is provided.

Chapter 5 reflects on the results obtained in the previous chapters, and offers a more elaborate discussion and interpretation, examining several implications and potential issues arising from this study. These considerations are evaluated as potential starting points for further research; in the final part, this chapter presents the many possible avenues for future research that are opened up by this study.

Finally, chapter 6 summarizes the contributions of this dissertation and discusses future work currently being developed.

1.6 Publications

The results of the research presented in this dissertation have been the object of the following publications:

- F. M. Zennaro, K. Chen. Towards Understanding Sparse Filtering: A Theoretical Perspective. *Under review*, 2017. Available at: https://arxiv.org/abs/1603.08831.
- F. M. Zennaro, K. Chen. On Covariate Shift Adaptation via Sparse Filtering. *Under review*, 2017. Available at: https://arxiv.org/abs/1607.06781.
- F. M. Zennaro, K. Chen. Covariate Shift Adaptation via Sparse Filtering for High-Dimensional Periodic Data. In *NIPS 2016 Workshop on Learning in High Dimensions with Structure*. Barcelona, Spain, 2016.

The first article (Zennaro and Chen, 2016a) covers the theoretical and empirical analysis of SF presented in Chapter 3, while the second article (Zennaro and Chen, 2016b) deals with the theoretical and empirical study of CSA discussed in Chapter 4. The last workshop paper discusses some of the theoretical and empirical results in Chapter 4.

Chapter 2

Background

This chapter provides an introduction to this dissertation starting with an overview of the field of machine learning and concluding with a closer review of the specific topics that will be studied in the following chapters. The aim is to sketch a map of the research field clear enough to show where the subjects of this dissertation, that is FDL and CSA, belong and how they relate with other topics in the wider field.

Section 2.1 presents the reader with an intuitive introduction to the problem of learning, in which we attempt to capture the types of problems with which machine learning is concerned and to define in an unambiguous way the informal terms used throughout this dissertation to explain and interpret results. Section 2.2 moves on to review how these ideas are rigorously formalized within the framework of machine learning. Section 2.3 and Section 2.4 focus on the two specific current areas of research in machine learning which are the topic of this work, FDL and CSA respectively. Section 2.5 concludes by explaining the challenges and the open problems related to FDL and CSA.

2.1 The Problem of Learning

This section introduces the problem of learning. Consistently with our methodology, a top-down approach is adopted to describe the problem of learning. Section 2.1.1 starts with a an informal and intuitive presentation of the problem of learning: *what is it meant by learning*? *what are the features of learning that we are interested in*? Finally, Section 2.1.2 moves on to the more formal plane of machine learning wondering: *how can the specific idea of learning in computational terms be formalized*?

2.1.1 Intuitive description of the problem of learning

Learning is a generic term used to denote the ability of producing or accumulating knowledge from experience. This ability is often correlated with the idea of intelligence, in the sense that learning efficiently and correctly has been often seen as one of the hallmark of intelligence. Learning, however, may be related with other distinct high-level faculties beyond intelligence,

such as memory, processing of information, modelling, observation making, reasoning and experimenting. To try to capture the meaning of learning, we may start with a very intuitive definition of LEARNING as: (1) a faculty of different organisms that allows for complex and refined interaction with an environment[1]. It may be said that learning implies the ability to capture regularities and laws that underlie an environment. Extracting regularities allows for a better understanding of the environment, leading to refined decision making and sophisticated planning, and enabling the potential of prediction and control over the same environment.

Trying to narrow our focus by locating the faculty of learning, we may re-define it as (2) a loose set of mental activities that account for the accumulation of knowledge useful for interacting with an environment. Now, this loose set of mental activities may be studied from a variety of points of view; for instance, it may be studied from a material-neurophysiologic point of view (*how are these processes implemented?*), from a diachronic-evolutionary point of view (*how did these processes evolve?*), or from a quantitative-economic point of view (*which differential advantages do these processes confer?*). In the following, we will adopt a functional-engineering point of view, mainly concerned with the question of how the processes underlying learning can be modelled and, eventually, replicated. This approach allows us to ignore the actual nature of the learning processes and deal with it only in abstract terms.

When talking of learning as a process, we want to underline that what we are concerned with is the actual process of the generation of knowledge. We may then refine the definition of learning in terms of LEARNING PROCESS as (3) a loose set of mental activities through which an agent is able to transform data provided by experience into new knowledge useful for interactions with an environment. This definition contains all the basic elements necessary to describe learning as a generic process in engineering terms; indeed, it specifies the *inputs*, *outputs*, *aim* and the *transformation* of a learning process:

- *Input of a learning process:* the input (or the object) of the learning process is DATA and INFORMATION. We assume the definition of *datum* as a measurable difference due to the lack of uniformity in the reality or in a signal being processed by an agent (Floridi, 2011). Similarly, we assume the definition of *information* as a collection of data that are well-formed, meaningful and truthful (Floridi, 2011). Since in the following we will only deal with data that are well-formed (formatted for computational processing), meaningful (having values in a consistent domain) and truthful (excluding the possibility of tampering by adversary agents), we will use data and information interchangeably. Notice that defining learning as capturing regularities makes perfect sense when the object of learning is data or information defined as a lack of uniformity; indeed, it is this lack of uniformity that generates the patterns that are the object of learning.
- *Output of a learning process:* the output (or the product) of the learning process is KNOWLEDGE. Defining knowledge is more challenging and is a matter of debate. For the sake of this work, we will assume knowledge to be an upgraded and relevant form of

[1] For comparison, the Oxford English Dictionary defines learning as: "The action of [...] acquiring knowledge; [...] a process that leads to the modification of behaviour or the acquisition of new abilities or responses. [...]".

information (Floridi, 2011). In general terms, we require knowledge to explain data, that is, to describe data synthetically and in a way relevant for making predictions or taking decisions. This sort of knowledge can be expressed using MODELS. The concept of model is a very generic and powerful concept (Floridi, 2011), and here we define it as a simplified and quantitative abstraction of an aspect of reality; within given conditions, we expect this abstraction to be in direct correspondence with the reality itself. As such, a model is meant to capture and describe a regularity, a pattern or a law underlying the reality with the aim of describing, predicting, controlling or reproducing it (Floridi, 2011). A model can usually encode laws at different levels of abstraction (Floridi, 2011): there is often a trade-off between how simple a model is and how close it corresponds to its real counterpart. Simpler models may constitute a gross approximation of what we are trying to learn, while more complex models may come closer to a more precise description of the pattern in the data. The activity of modelling has been at the core activity of many scientific and experimental disciplines and it has been often identified with the activity of learning itself.

- *Aim of a learning process:* the aim (or the purpose) of the learning process is to provide knowledge that is useful for interacting with an environment. Practically, it is often assumed that the knowledge produced by the learning process is either meaningful, useful or somehow usable. Learning is taken to be inherently *purposeful*. There are, in other words, implicit principles (utilitarian, logical, aesthetic) that drive the learning process from the data to certain forms of knowledge instead of others. The quality of learning is often defined in relation to these external aims set by a learning agent or constrained by the environment; for instance, meaningfulness may be evaluated with reference to an external framework of meanings, usefulness may be measured in relation to an external goal, usability may be measured as a function of fitness in an environment.

- *Transformation of a learning process:* the transformation of the learning process defines how the input is converted into the output consistently with the chosen aim. The type of transformation is intimately connected to the type of information the learning process is applied to and the type of knowledge it aims at producing. With reference to classical theories of logic, it is possible to distinguish (at least) three main forms of learning:

  - DEDUCTION: Deduction is defined as a form of logical reasoning through which an agent derives *general or particular conclusions* from *general premises*. Information consists of a set of general premises; knowledge is constituted by another set of general or particular conclusions. For instance, an agent may derive mathematical theorems from a set of axioms.

  - INDUCTION: Induction is defined as a form of logical reasoning through which an agent derives *general conclusions* from *particular premises*. Information consists of a collection of particular data; knowledge is constituted by a set of general conclusions. For instance, an agent may collect specific observations about a phenomenon of interest and derive a general law to explain it.

– TRANSDUCTION: Transduction is defined as a form of logical reasoning through which an agent derives *particular conclusions* from *particular premises*. Information consists of a collection of particular data; knowledge is constituted by a set of conclusions regarding the same data. For instance, an agent may collect specific observations about a phenomenon of interest and derive conclusions that explain those specific instances she observed.

In terms of data and information, PARTICULAR PREMISES or PARTICULAR CONCLUSIONS can be described as finite, noisy and low level forms of information. FINITE INFORMATION means that a learning agent will inevitably have just a limited set of data; this set of data is taken to be a particular product, not a complete and exhaustive description, of a general rule. NOISY INFORMATION means that not all the pieces of information provided are relevant for the current aim of learning; data may be affected by intrinsic noise (biases introduced by the instruments used for data collection) or semantic noise (data not related to the current aim of learning); clearly, noise is a relative concept, defined and evaluated in relation to the purpose of learning, and any sort of non-relevant information that clutters the data can be taken to be noise. LOW-LEVEL INFORMATION means that the information is bare data directly derived from human senses or artificial sensors, implying no refined processing; obviously, low-level, too, is a relative concept, as any sense or any sensor is hard-coded to perform some processing; low-level is mainly used to denote a given starting state which is considered unsuitable for a specific purpose at hand.

By contrast, GENERAL PREMISES or GENERAL CONCLUSIONS may be described as generic, useful and high-level forms of information. GENERIC INFORMATION means that knowledge can explain not only a finite set of data generated according to some rule, but every possible set of data generated using the same rule. USEFUL INFORMATION means that knowledge satisfies the purpose of learning, thus complying with requirements of meaningfulness or utility for a learning agent; like noise, usefulness is a relative term and it is evaluated in relation to a specific aim. HIGH-LEVEL INFORMATION means that knowledge provides a non-trivial understanding or insights; this information must be implicitly present in the original data, but not explicitly available for use before learning; high-level, then, is mainly used to denote a final ideal state fit for specific purposes.

Historically, the study of deduction as a form of learning has been the domain of traditional logic and classical artificial intelligence. Statistics and standard machine learning, instead, have been more concerned with the use induction and transduction in order to draw conclusions from finite sets of data. Transduction can actually be seen as a form of learning subsumed by induction. Indeed, by restricting the validity of the output of induction from a generic set of data to the specific available set, induction is reduced to transduction. Thus the conclusions of induction subsume the conclusions of transduction, even if the reverse does not normally hold. For this reason, from now on, whenever we will refer to the learning problem, we will implicitly mean an inductive learning process, which we can now define as (4) a process in which finite, noisy and low-level data are transformed into generic, useful and high-level knowledge in order to pursue an aim.

2.1.2 Computational description of the problem of learning

So far, we have discussed an intuitive conceptualization of learning. However, to better understand the process of learning and to be able to replicate it, we need to make this description of learning more rigorous.

Given that the learning process deals with transformation of information, it seems particularly suitable to express it in *computational terms*. We therefore make the following assumption:

Assumption (Computational Model). *Inductive learning can be reduced to a deterministic computable process of information processing.*

This assumption may be debated by questioning whether all forms of inductive learning can be reduced to information processing or whether they all constitute forms of deterministic processing, but here we will take it for granted. This is, after all, a prominent position in artificial intelligence whose grand aim is to replicate human intelligence, and for which implementing computationally the faculty of learning represents a necessary step toward this objective. Within artificial intelligence the study of the computational problem of inductive learning has been the specific domain of machine learning and, for this reason, we will refer to the inductive learning process in computational terms as the problem of machine learning or as a MACHINE LEARNING PROCESS.

Based on the above assumption, we can then try to express the learning process in abstract computational terms, so that learning can then be implemented on any sort of device able to carry out computations. To translate the problem of learning in precise computational terms, it is necessary to provide a more rigorous and quantitative way to define the elements of the learning process, that is its inputs, outputs, aim and transformation. Notice that these new definitions need to be not only quantitative, but computable, that is, they must be quantitative and representable on a digital machine.

- *Input of a machine learning process:* data and information can be formalized through the concept of REPRESENTATION. A representation is essentially a computational tool, a way to shape data and information in a form that can be processed computationally (Grosse, 2014). Representation is then defined as any discrete quantitative collection of data and information regarding a phenomenon of interest. In particular, we will call ORIGINAL REPRESENTATION a set of data provided as an input to the machine learning process. During the learning process, these representations may be manipulated and changed; we will call the transformed representations LEARNED REPRESENTATIONS; the last learned representation produced by the machine learning process is assumed to encode some form of knowledge extracted during the learning process. Notice that the adjectives "original" and "learned" are relative to a specific learning process and they are adopted simply for convenience, in order to point out the starting and the ending point of a learning process. As noise and low-level information are relative concepts, the learned representation of a machine learning process may immediately become the original representation of another computational learning process.

- *Output of a machine learning process:* knowledge in the form of a model may be represented through ALGORITHMS. An algorithm is defined as a strict list of deterministic instructions which can be implemented by a Turing machine (Turing, 1937). An abstract algorithm may theoretically implement any computable function. Thus, as long as the model we are learning can be expressed as a computable function, it can be expressed and encoded in the form of an algorithm.

- *Aim of a machine learning process:* the aim of a learning process can be formalized through the concept of UTILITY. Utility is defined as a quantitative measure of how well a machine learning process achieves its purpose. We will use the utility (or an approximation of it) as a numerical value to drive the machine learning process and to assess the goodness of a learned model. Utility is externally defined by the agent implementing the machine learning process, and it may take into consideration different criteria for evaluation, such as usefulness, complexity, efficiency, redundancy or interpretability of the learned model.

- *Transformation of a machine learning process:* the transformation, or more specifically, the induction of a learning process can be formalized again through the concept of an algorithm. However, it is important to underline that while earlier we talked of algorithms as the outcome of the learning process, here we refer to algorithm as the actual implementation of the learning process that leads to learning a model. To make things clearer, we may refer to the algorithm that implements the induction process as a META-ALGORITHM. Thus, we can say that the transformation of a learning process is implemented as a meta-algorithm that can learn a specific algorithm-model.

Based on this formalization we can sum up the definition of a machine learning process as (5) a (meta-)algorithm processing data representations and producing an algorithm(-model) consistent with a given utility.

The interpretation of a machine learning process in these terms provides us with a high degree of flexibility. By abstracting learning into a process, it is possible to compose and decompose complex forms of learning into simpler processes that can be combined together. Following a simple rule of compositionality, as long as the output of a learning process matches the input of another one, it is possible to easily connect together different simple learning processes, with potentially different aims and transformations, into more complex learning processes.

In conclusion we can summarize this first understanding of learning and its computational conceptualization as illustrated in Table 2.1.

Now that we have explored what is meant by learning and how it can express the inductive problem in computational terms, we will move on to analyse more closely how machine learning sets up the problem of learning.

Process	***Learning as an intuitive process***	***Learning as inductive process***	***Learning as transductive process***	***Learning as a machine learning process***
Input	Data (information)	Finite, noisy, low-level data	Finite, noisy, low-level data	Representations
Output	Knowledge (information)	Generic, useful, high-level knowledge	Finite, noisy, low-level data	Model (algorithm)
Aim	Interaction with environment	Interaction with environment	Interaction with environment	Utility
Transformation	Deduction / Induction / Transduction	Induction	Transduction	Algorithm (meta-algorithm)

Table 2.1: Definition of learning in terms of process: from the intuitive understanding to the machine learning understanding.

2.2 Machine Learning

The previous section described the problem tackled by machine learning as the implementation of algorithms that, starting from computable representations of data, produces an algorithm which encodes a model explaining the data according to a specific aim. This section overviews the details of how machine learning concretely implements inductive learning. Section 2.2.1 starts by presenting the *languages* used by machine learning to define data representations and algorithms. Section 2.2.2 gives a brief overview of the *criteria* used to make design choices in machine learning. Based on these notions, Section 2.2.3 explains how machine learning algorithms are concretely defined using the presented languages and the design criteria. Finally, Section 2.2.4 provides a taxonomy and a brief description of the most important machine learning problems confronted in the literature, which is used as a reference to place the topics of this dissertation in context.

2.2.1 The languages of machine learning

In order to express the problem of machine learning in computational terms, one or more rigorous and consistent languages to formalize the notions of representation, information and algorithms are needed. Machine learning traditionally relies on the language of other mathematical disciplines. Beside the general language of calculus and computer science, machine learning specifically borrows concepts from the fields of linear algebra, statistics, information theory and optimization. Adopting concepts from these diverse fields brings several benefit to the study of machine learning: it makes it possible to rely on rigorous definitions, import results, exploit tested frameworks and borrow interpretations and explanations. Linear algebra, statistics, information theory and optimization offer well-integrated mathematical languages that constitute a solid foundation upon which machine learning can incrementally build its content.

The following sub-sections briefly recall the basic concepts that machine learning takes from each field. We will take these basic concepts for granted and not discuss them in detail. We will, however, explain how they are applied to the problem of learning.

2.2.1.1 Linear algebra language

Linear algebra provides a solid and well-develop framework to describe data and their transformations in high-dimensional spaces. Machine learning borrows from linear algebra the concepts of vector spaces, matrices, linear transformations and morphisms.

When adopting the language of linear algebra to model the problem of machine learning, we implicitly make the following assumptions.

Assumption (Matrix representation). *Original and learned representations can be properly represented as a set of deterministic quantitative values encoded in matrix form.*

Assumption (Morphisms). *A model can be expressed as a morphism between spaces, usually vector spaces.*

The first assumption means that each representation can be considered as an observation made up of a collection of quantitative values called FEATURES. This assumption is strictly related to the essentialist approach to machine learning (Pelillo and Scantamburlo, 2013), which asserts that observed entities have essential properties defining them. The second assumption identifies the computational transformation to be learned as a generic morphism.

More rigorously, let X be the set of original representations. We encode X as a matrix $\mathbf{X}$ containing $N \in \mathbb{N}_{>0}$ observations. Each observation $\mathbf{x}_i$, $1 \leq i \leq N$, is defined on the vector space $\mathcal{X} \subseteq \mathbb{R}^M$ as a domain, and it is encoded as a vector with $M \in \mathbb{N}_{>0}$ dimensions. Each dimension $\mathbf{x}_{\cdot,j}$, $1 \leq j \leq M$, corresponds to a feature defined on a domain that, for the sake of generality, we assume to be $\mathbb{R}$. Thus, $\mathbf{X}$ is a matrix defined[2] on $\mathbb{R}^{M \times N}$.

Analogously, let Z be the set of learned representations. We encode Z as a matrix $\mathbf{Z}$ containing the same number N of observations. However, each observation $\mathbf{z}_i$, $1 \leq i \leq N$, is now defined on a vector space $\mathcal{Z} \subseteq \mathbb{R}^L$ as a domain, and it is encoded as a vector with $L \in \mathbb{N}_{>0}$ dimensions. Again, each dimension $\mathbf{z}_{\cdot,j}$, $1 \leq j \leq L$, corresponds to a feature defined, for the sake of generality, on $\mathbb{R}$. Thus, $\mathbf{Z}$ is a matrix defined on $\mathbb{R}^{L \times N}$.

A model is a morphism $f : \mathcal{X} \to \mathcal{Z}$ chosen from the space of morphisms $\mathfrak{F}$ which projects representations from the original space $\mathcal{X}$ into the learned space $\mathcal{Z}$.

The language of morphism implicitly grants us the property of compositionality according to which a learning process can be decomposed into several simpler processes combined together. Indeed, the property of compositionality of morphisms allows us to apply several functions in series as long the co-domain of one function corresponds to the domain of the following function: $f'' \circ f'(\mathbf{X}) = f''(f'(\mathbf{X}))$, where $f' : \mathcal{X}' \to \mathcal{Z}'$, $f'' : \mathcal{X}'' \to \mathcal{Z}''$ and $\mathcal{Z}' = \mathcal{X}''$. Practically, this

[2] Notice that, for consistency with Ngiam et al. (2011), we adopt the slightly unconventional notation where the features are defined on the rows and the samples along the columns.

gives us the possibility of stacking or pipelining different machine learning models into complex systems.

In sum, in the formalism of linear algebra, learning amounts to choosing a learned space $\mathcal{Z}$ where we can encode high-level and useful information and discovering a morphism $f \in \mathfrak{F}$ that leads from the finite, low-level and noisy information in the original representation $\mathbf{X}$ to high-level and useful information.

2.2.1.2 Statistics language

Statistics provides a rigorous and quantitative way to deal with uncertainty in the learning process. Machine learning borrows from statistics the concepts of random variables, probability distributions, probability density functions (pdf), probability mass functions (pmf), cumulative distributions, marginal distributions, joint distributions, conditional distributions and statistical moments.

The language of statistics allows us to approach the problem of learning from two radically different perspectives, in relation to where we assume uncertainty to lie.

Assumption (Frequentist approach). *Data are samples from a probability distribution with fixed parameters.*

Assumption (Bayesian approach). *Data are fixed quantities generated by a process whose parameters are modelled as probability distributions.*

These two assumptions lead to two perspectives on machine learning that are radically different: the frequentist approach sees uncertainty mainly as an effect of the limited number of samples, and it aims for the best estimation of the fixed parameters of the probability distribution that generated the available samples; the Bayesian approach sees uncertainty as an intrinsic condition of our knowledge about the parameters that govern the process that generated the data, and it aims at modelling these parameters through probability distributions. Even though the results of the two approaches are consistent, they employ a range of subtly different conceptual tools, such as parameter estimation versus distribution estimation, hypothesis testing versus hypothesis comparison (MacKay, 2003), confidence intervals versus credible intervals (Jaynes and Kempthorne, 1976). However, analysing the conceptual difference between these two approaches is beyond the scope of this thesis (for a brief discussion, see, for instance, Efron, 2005). In this dissertation a frequentist perspective is adopted.

When adopting the language of statistics with a frequentist point of view in order to model the problem of machine learning, we implicitly make certain assumptions.

Assumption (Sample representation). *Original and learned representations can be properly represented as a set of stochastic quantitative values encoded as samples from a random variable.*

Assumption (Probabilistic Manifolds). *A model can be expressed as a transformation on manifolds of pdfs (Amari, 2016).*

These two assumptions mirror the equivalent assumptions made for the formalism of linear algebra.

Formally, let X be the set of original representations. We formalize X as a collection $\mathbf{X}$ of $N \in \mathbb{N}_{>0}$ samples generated by a multivariate random variable X with a marginal pdf $p(X)$. Analogously, the set of learned representations Z is formalized as a collection $\mathbf{Z}$ of the same number N of samples generated by a multivariate random variable Z with a marginal pdf $p(Z)$. The overall behaviour of original representations and learned representations is explained by the joint pdf $p(X, Z)$, while the behaviour of the learned representation given the original representations is explained by the conditional pdf $p(Z|X)$.

Given the samples $\mathbf{X}$ and $\mathbf{Z}$, statistics provides methods to estimate the generating pdfs, and evaluate the sample marginal pdfs, $\hat{p}(X)$ and $\hat{p}(Z)$, the sample joint pdf, $\hat{p}(X, Z)$, and the sample conditional pdf, $p(\hat{Z}|X)$. Ideally, knowledge of the pdf of a random variable X gives a perfect description of X. However, the estimation of a pdf in a high dimensions with a limited amount of data is often challenging and unreliable (Bishop, 2007). Practically, the difficulty in estimating a pdf often leads to the adoption of simpler synthetic indexes to meaningfully summarize a pdf with a single scalar value or with a small collection of scalar values.

The most intuitive and common descriptors for a pdf are its statistical moments. The 0-th moment $M_0[X] = \int_{\mathcal{X}} x^0 \cdot p(X = x)\, dx$ of a pdf $p(X)$ defines its volume (which is actually a not very useful descriptor, since every pdf has a volume of 1); the 1-st moment $M_1[X] = E[X] = \int_{\mathcal{X}} x \cdot p(X = x)\, dx$ is its expected value (which defines the barycentre of the realizations of X); the 2-nd moment $M_2[X] = \int_{\mathcal{X}} x^2 \cdot p(X = x)$ is related to the variance $Var[X] = E\left[X^2\right] - E[X]^2 = E\left[(X - E[X])^2\right]$ (which gives an index of the spread of the realizations of X); higher moments provide further information about the shape of $p(X)$ (such as its skewness or kurtosis). A finite amount of moments provides a usually coarse approximation of a pdf, unless other constraints fixing the values of higher moments are available (for instance, in the case of a Gaussian pdf all the moments above the second are zero, therefore knowing the mean and the variance gives a complete description of the pdf). A complete description of a pdf is given by the knowledge of all its moments, which, in general, means an infinite number of values.

As in the case of pdfs, statistical moments can be estimated from samples $\mathbf{X}$ and $\mathbf{Z}$. Estimation of lower statistical moments, such as sample expected value $\hat{E}[X] = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$ or sample variance $\hat{Var}[X] = \frac{1}{N-1}\sum_{i=1}^{N}\left(\mathbf{x}_i - \hat{E}[X]\right)^2$ is usually more reliable than the estimation of the whole pdf, and statistics provides properties of consistency and bounds on their estimation. Therefore, the estimation of these simple descriptors is a common and widespread tool to analyse pdfs.

Now, the formalism of linear algebra and the formalism of statistics integrates together, providing complementary perspectives on the same problem of inductive learning. In both cases, the aim of learning is to produce a model that implements a map from an original space $\mathcal{X}$ to a learned space $\mathcal{Z}$. It is immediate to move from one formalism to the other. The same

Machine Learning Concept	***Linear Algebra Formalism***	***Statistics Formalism***
Input data	$\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^N$	$\mathbf{x}_i \sim p(X)$
X	$\mathbf{X} \in \mathbb{R}^{M \times N}$	$\mathbf{X} \in \mathbb{R}^{M \times N}$
Intermediate data	$\mathbf{z}_i \in \mathcal{Z} \subseteq \mathbb{R}^L$	$\mathbf{z}_i \sim p(Z)$
Z	$\mathbf{Z} \in \mathbb{R}^{L \times N}$	$\mathbf{Z} \in \mathbb{R}^{L \times N}$
Output	$f : \mathcal{X} \to \mathcal{Z}$	$f : \mathcal{X} \to \mathcal{Z}$
Aim	Utility	Utility
Transformation	Algorithm	Algorithm

Table 2.2: Linear algebra and statistics formalism to describe the machine learning problem.

syntax allows us to switch easily between the two views: $\mathbf{X}$ can be interpreted as a fixed matrix or as a set of realizations of a random variable, and $\mathbf{x}_i$ can be interpreted as a fixed vector or as a random vector (see Table 2.2). The algebraic formalism highlights the conception of learning as the processing of deterministic signals and the discovery of patterns. The statistical formalism underlines the idea of learning as the definition of stochastic models that can fit complex, non-deterministic phenomena. In the following, we will interchangeably refer to the two formalisms in order to offer complementary explanations and interpretations of our results.

2.2.1.3 Information theory language

Information theory provides a versatile framework to deal with the notion of information. Machine learning borrows from information theory the concepts of entropy, mutual information and relative entropy.

The adoption of the language of information theory is dependent on a strong assumption about the concept of information.

Assumption (Syntactic information). *Information, as a quantitative parameter, does not measure the content or the meaning of the data, but a difference between how the data is expected to distribute and its actualized distribution.*

The amount of information measured through entropy is not related to the semantics of the data, but only to its statistical behaviour. A channel transmitting with equal probability zero or one has the same amount of information as a channel transmitting with equal probability one half of Wikipedia or the other (Frigg and Werndl, 2011). This conception of information may be explained by the intuitive idea of unexpectedness or surprise: data are informative when they strike us as unexpected. Information describes then the degree of confidence in a set of potential events. A set of data provides information if it affects the degree of confidence in the set of events. Notice that, even if the events are necessary and deterministic, it is possible to describe their occurrence, in relation to an observer's knowledge, as stochastic events.

As discussed above, if we model representations as realizations of a random variable X, a complete description of this random variable would be provided by its pdf. As a pdf is hard to estimate and to deal with, we rely on descriptors, such as the statistical moments. Information theory provides alternative and more robust measures to describe a pdf (Principe, 2010). In particular, entropy is a generic statistical measure for pdfs which formalizes the concept of information, or, better, the concept of unexpectedness, which we described above.

Given the pdf $p(X)$, we quantify the information contained in $p(X)$ as the Shannon's entropy $H_S[X] = -\int_{\mathcal{X}} p(X=x) \log p(X=x)\, dx$ (MacKay, 2003). This entropy denotes how close the pdf $p(X)$ is from an entropy-maximizing pdf $q(X)$ on the same domain $\mathcal{X}$. Thus a pdf $p(X)$ with low entropy is far from $q(X)$ and it allows us to localize with high confidence its possible realizations on a restricted subspace of $\mathcal{X}$. On the other hand, a pdf $p(X)$ with high entropy is close to $q(X)$ and it forces us to accept many possible realizations of the random variable X.

Notice that the description of entropy in terms of potential values assumed by the random variable X puts it in close relation with variance. Even if entropy and variance may often agree in the ordering of pdfs according to their dispersion, it can be shown through a Legendre series expansion that entropy carries information related to higher moments beyond the second; as such, entropy is more descriptive than the simple variance (Ebrahimi et al., 1999). Actually, entropy is a fundamental scalar descriptor of a pdf $p(X)$, more powerful than other statistical moments, because it provides an accurate evaluation of the volume occupied by the pdf $p(X)$ and its realizations, as illustrated through the *asymptotic equipartition property* (Principe, 2010).

Given two pdfs $p(X')$ and $q(X'')$, it is possible to compute their relative entropy, or Kullback-Leibler divergence, $D_{KL}[p(X') \parallel q(X'')] = \int_{\mathcal{X}} p(X'=x) \log \frac{p(X'=x)}{q(X''=x)} dx$, in order to measure how close or similar the distributions are. A low distance means that the pdfs are similar, while a high distance denotes a difference. The Kullback-Leibler divergence was proved to possess several useful properties, but many alternative distances may be devised to estimate distances between pdfs (Kapur, 1994; Amari, 2016).

A measure to quantify the relationship between two random variable X' and X'' is the mutual information $MI[X'; X''] = \int_{x \in \mathcal{X}} \int_{y \in \mathcal{X}} p(X'=x, X''=y) \log \frac{p(X'=x, X''=y)}{p(X'=x)p(X''=y)} dxdy$, which measures the shared information between X' and X'' and how much can be learned about one given the other. By definition, mutual information is related to the Kullback-Leibler divergence of the joint pdf and the product of the marginals of $p(X')$ and $q(X'')$: $MI[X'; X''] = D_{KL}[p(X', X'') \parallel p(X')\, p(X'')]$. Notice that Kullback-Leibler divergence and mutual information evaluates two different types of quantities: the Kullback-Leibler divergence is a measure of distance between pdfs (which may be changed by shifting the domains of the pdfs), while the mutual information gives an index of correlation between random variables.

As in the case with statistical descriptors, it is possible to compute estimation of these information-theoretic quantities when provided with samples $\mathbf{X}'$ and $\mathbf{X}''$. We can evaluate sample entropy $\hat{H}_S[X']$, sample relative entropy $\hat{D}_{KL}[p(X') \parallel q(X'')]$, or sample mutual information $\hat{MI}[X'; X'']$. Inevitably, there is a cost in estimating this more informative descriptors, in that their estimation is more difficult and challenging than the estimation of

simpler statistical moments.

It is under the understanding of information as a syntactic property (see assumption above) that information-theoretic descriptors may be used to direct the process of learning. If knowledge is encoded in a random variable or in a pdf, it makes sense to rely on information-theoretic measures to manipulate the representations. Minimizing the entropy of a random variable equates to reducing uncertainty about its outcomes; minimizing or maximizing the relative entropy between two pdfs forces them to behave in similar or different ways; maximizing the mutual information between two random variables, such as data X and labels Y, allows for an easier guess or inference about one from the other. It is important, though, to underline the limits of a formal analysis through information-theoretic quantities and the necessity to precisely define the aim of learning. The *data processing theorem* (MacKay, 2003) states that data processing can only destroy information, that is, given two random variables X and Y, then $MI[X;Y] \geq MI[f(X);Y]$, for any function $f()$. This theorem seems to lead to a paradox: on one side, it makes sense to state that it is not possible to create information but only to rely on what it has been provided by the data; on the other side, however, by preventing the possibility of increasing information, it seems to defeat the understanding of learning as a pure optimization of information-theoretic quantities. This paradox is solved by noticing that this apparent conflict is simply a consequence of the syntactic definition of information; it is syntactic information in the form of theoretical uncertainty and probability that cannot be improved, not the semantic form as relevant knowledge or availability. Data processing, then, still makes sense in terms of a semantic *information distillation* (Lin and Tegmark, 2016), where syntactic information may be lost in the form of removal of noise and simplification of the model explaining the data.

Finally, it is easy to see how the language of statistics and information theory seamlessly integrate together. If we decide to model a machine learning problem with the language of statistics, then information theory provides additional tools to study and to manipulate pdfs. So it is possible at the same time to analyse a pdf using a collection of elementary statistical moments and to evaluate its shape in terms of entropy.

2.2.1.4 Optimization language

Optimization theory provides effective and efficient methods to define both the aim of learning and the way to pursue this aim. Machine learning borrows from optimization theory the concepts of maximization/minimization, optimization, convex optimization and gradient descent.

The adoption of the language of optimization is based on the following simple assumption.

Assumption (Quantitative Objective). *The aim of learning can be measured.*

This assumption is necessary in order to deal with learning in the quantitative terms of mathematics. Not surprisingly, it sets a clear limitation on what can be learned through machine learning: learning can happen if and only if an aim can be expressed in measurable terms.

Defining the aim of learning through the language of optimization theory means translating it into an optimization problem. An optimization problem, then, evaluates potential learned models f with respect to the given objective. To do so, we define an objective function, rigorously a functional, $\mathcal{L} : \mathfrak{F} \to \mathbb{R}$ that assigns to each possible discoverable function f in $\mathfrak{F}$ a real value corresponding to its goodness. The objective function $\mathcal{L}$ is called the *utility function*, if its value increases as the quality of the solution f improves, or the *loss function*, if it decreases as the quality of the solution improves. For simplicity and consistency with the literature, we will now refer to an objective function as a loss function.

Optimization theory defines standard approaches to solve optimization problems. It allows us to re-cast the problem of learning as a *searching problem* (learning a model f equates to discovering the right model f within the set of possible models $\mathfrak{F}$) or as a *functional optimization problem* (learning a model f equates to minimizing a loss functional $\mathcal{L}$ as a function of f). Optimization theory and operational research provide a vast array of techniques to solve functional optimization problems, ranging from computationally efficient algorithms for solving easy optimization problems to expensive techniques for finding approximate solutions to challenging problems.

The choice of a loss function and the choice of an optimization algorithm are intimately connected. In making these two choices we often have to balance two opposite desiderata: (i) we want a loss function to be meaningful; (ii) we want a loss function that can be efficiently optimized. Meaningfulness means that the loss function closely reflects our idea of how good a model is, returning low values for models we would consider good and returning high values for models that we would reject. Efficiency means that the loss function is easily tractable, thus making the search within the space $\mathfrak{F}$ fast and efficient. Usually, there is a trade-off between these two desiderata, as very meaningful loss functions may often be too difficult to optimize. Inevitably, it is often necessary to compromise meaningfulness for efficiency, or vice versa.

2.2.2 Criteria for design choice in machine learning

So far, we have described the languages used by machine learning and the main concepts applied to the modelling of the machine learning problems. However, when we come to designing specific instances of learning algorithms, we are required to make several modelling choices. We face here a meta-problem: how are we going to tackle and solve specific problems that will arise in formalizing a machine learning problem? To answer this question we explore here criteria to which we can refer to whenever we have to make such a choice.

2.2.2.1 Principles

Whenever we are called to make a modelling choice, we may appeal, at the highest level, to generic PRINCIPLES. Principles are abstract criteria with a wide epistemological validity.

Examples of well-known principles which will be referred to later include:

Principle (Laplace's principle of insufficient reason). *Two events should be assigned equal probabilities if there is no reason to think otherwise (Jaynes, 1957).*

This is a principle of indifference stating that, without proper information, an outcome should not be favoured or weighted more than another one.

Principle (Maximum entropy principle). *In modelling a distribution, select the probability distribution that has maximum entropy subject to whatever is known (Jaynes, 1957).*

This principle generalizes the principle of insufficient reason, stating that, without proper information, uncertainty should not be arbitrarily reduced.

Principle (Occam's razor). *Accept the simplest explanation that fits the data (MacKay, 2003).*

Occam's razor is a widespread principle in science which suggests to opt for simpler models, *ceteribus paribus*.

Principle (Generalization). *An inductive model must not explain only the data from which it is learned, but it must explain any possible data generated in the same way.*

This is a central principle in machine learning, and it is actually a direct consequence of the definition of induction.

Principle (Preservation of data). *Data should not be discarded without valid reasons.*

This is an important principle in statistics, both for economical reasons (samples are expensive to collect and carry exploitable information) and for theoretical reasons (rejecting samples could introduce sample selection bias in the learning process, Heckman, 1977).

Of course, these principles may be the object of debate. However, such a discussion is outside the scope of this dissertation, and we will take these principles as self-evident.

2.2.2.2 Assumptions

On a more concrete level, we may have to make choices about our meta-models and models, and their specific relationship with the modelled entity. This includes defining at which level of abstraction we want to work, how much simplification or complexity we want to consider and what aspects of reality our model is going to capture. In this case, we may rely on ASSUMPTIONS (MacKay, 2003): specific and arbitrary choices that define the relation of correspondence between our model and the real phenomenon we are considering. Assumptions may define what aspects of a phenomenon under study are relevant for the analysis, what aspects can be disregarded and which potentially interesting relationships we want to uncover.

Similarly to axioms in a deductive framework, assumptions are taken to be true at the beginning. Assumptions define the limits of the correspondence of the model to reality. Whenever the assumptions underlying the model are violated, the correspondence with reality cannot be guaranteed. As different axioms give rise to different deductive systems, so different assumptions generate different inductive models. Inductive models grounded on different assumptions,

like deductive systems founded on different axioms, constitute descriptions of a phenomenon at different levels of abstraction, which are not correct or incorrect in absolute terms, but which may adhere more or less to reality according to our requirements.

In designing a machine learning algorithm we are inevitably bound to make assumptions (MacKay, 2003). Several assumptions lie behind every algorithm, ranging from trivial to more delicate and sophisticated assumptions. If we conceive of the process of designing a learning machine as a stochastic process, our choices should be guided by the principle of maximum entropy; assumptions are the elements of information that set the bounds and shape a probability distribution for our choices, thus allowing us to distinguish between alternatives and make grounded choices. For instance, the choice of parameters for a model can be seen as a choice driven by assumptions: instead of considering a uniform distribution over all the possible discrete values that a parameter can assume, we restrict the possibility to a limited interval relying, for instance, on assumptions about the type of data from which we are learning or about the dynamics of the algorithms we are using. Assumptions may be more or less explicit, and are sometimes taken for granted and not spelled out. Uncovering implicit or hidden assumptions is fundamental in order to understand the limits within which a machine learning algorithm is expected to work.

2.2.2.3 Prior Knowledge

Finally, in designing a machine learning system we may rely on specific knowledge that we have about the problem at hand. In this case, we rely on PRIOR KNOWLEDGE, that is knowledge available to the modeller before learning.

Prior knowledge can be elegantly expressed in a Bayesian framework through priors, that is pdfs that express the distribution of probabilities before learning. Alternatively, prior knowledge can be injected in machine learning algorithms in other ways, such as through the choice of parameters for the model or the introduction of constraints and penalties in the loss function.

Notice that assumptions and priors differ mainly from an epistemological point of view: assumptions are something we take to be true, while priors are something we know to be true. However, from a practical point of view, their content and their implementation may be identical. The actual distinction between assumptions and priors is at most vague. For instance, the same piece of knowledge about linear separability may be either an assumption (when it is taken for granted at the beginning) or a prior (when it is known for certain about the data).

Also, the same piece of knowledge which for certain machine learning algorithms is an assumption or a prior, could be a conclusion for other algorithms. For instance, always referring to the example of linear separability, this piece of knowledge may be an assumption or a prior (when we act from the beginning as if the data were linearly separable) or a conclusion (if, without assuming it at the beginning, we realize from the results at the end that the data are linearly separable). Stating assumptions or priors is therefore important also to avoid the

fallacy of presenting a hidden assumption or a hidden prior as a conclusion.

2.2.3 Designing machine learning systems

Designing machine learning systems is an intellectually complex activity, one which entails the definition of a meta-model able to learn a model that can explain a set of data. Relying on the languages and the principles we have discussed, we now review more closely how a machine learning algorithm is implemented. This requires several steps: (i) defining a proper machine learning algorithm that can process data; (ii) defining how data are represented; (iii) choosing what type of representation we want to learn; (iv) formalizing the learning objective; (v) adopting an algorithm to pursue the objective; (vi) partitioning the data to properly achieve learning generalization; (vii) establishing a measure to evaluate the degree of success. We present all these steps sequentially, even if the process of design is a complex and organic process in which each step may be revisited more than once at different times.

2.2.3.1 Defining the algorithm

The central component of a machine learning system is, of course, a machine learning algorithm `A`. This algorithm is expected to process data and, once provided with an objective function, perform some form of meaningful learning. The dynamics of an algorithm `A` are usually specified by a set of hyper-parameters $\xi_j \in \Xi$ that determines its behaviour and, consequently, its results. Choosing a value or a range of values for these hyper-parameters ξ_j defines one or more concrete meta-models for learning.

2.2.3.2 Defining the space of representations

The language of linear algebra provides an immediate way to encode the input data $\mathbf{X}$ as a matrix over a vector space $\mathcal{X} \in \mathbb{R}^M$ and the output data $\mathbf{Z}$ as a matrix over a vector space $\mathcal{Z} \in \mathbb{R}^L$.

2.2.3.3 Defining the space of morphisms

The language of linear algebra also provides a formalization of a model as a morphism $f : \mathcal{X} \to \mathcal{Z}$, with f chosen from the space of all morphisms $\mathfrak{F}$. Now, in purely mathematical terms, the space $\mathfrak{F}$ of admissible functions may be an infinite and non-denumerable set of functions. This is intrinsically a problem when dealing with finite machines such as computers, which can instantiate only a finite set of discrete functions. A common method to discretise the space of admissible models is through PARAMETRIZATION. Instead of considering an infinite and non-denumerable set of models $\mathfrak{F}$, it is possible to restrict the attention to the set $\mathfrak{F}_\Theta$ of models that can be generated by varying a limited number of parameters θ_i in the set Θ. Reducing arbitrary functions to parametrized functions can be, in some cases, mathematically justified, taking the parameters to define a set of bases that can approximate functions to a desired degree; for instance, continuous and (infinitely) derivable functions may be approximated by polynomial bases through Taylor expansion; periodic functions may be approximated by sinusoidal bases through a Fourier transformation; continuous functions on a compact subset of $\mathbb{R}^n$ may be

approximated through the transformation function of a sigmoidal multi-layer perceptron (Cybenko, 1989). Through parametrization, the aim of learning is then reduced to the discovery of the optimal values of set of parameters Θ that specify a learnable model.

2.2.3.4 Defining the loss function

Having just defined the discoverable functions f as functions defined over a the set of parameters Θ, we can redefine the objective functional as $\mathcal{L} : \mathfrak{F}_\Theta \to \mathbb{R}$, that is a functional assigning a value to each possible function f that we can generate by the variation of the parameters $\theta_i \in \Theta$. Moreover, it is possible to reduce the loss functional to the function $\mathcal{L} : \mathbb{R}^{\|\Theta\|} \to \mathbb{R}$, that is a function that simply assigns a value to each possible vector of parameters $\theta_i \in \Theta$.

At learning time, the value of a loss functions $\mathcal{L}$ can be estimated from a set of data $\mathbf{X}$ and from the current selected model f. Recalling that a selected model is defined by its hyper-parameters ξ_j and its parameters θ_i, the empirical loss function $\hat{\mathcal{L}} : \mathcal{X} \times \mathbb{R}^{|\Xi|} \times \mathbb{R}^{|\Theta|} \to \mathbb{R}$ returns the estimated loss for the current selection of hyper-parameters ξ_j and parameters θ_i with data $\mathbf{X}$. Typically, the empirical loss function is estimated given the data, $\hat{\mathcal{L}}(\xi_j, \theta_i; \mathbf{X})$, since the data $\mathbf{X}$ cannot change during learning, differently from the hyper-parameters and the parameters. In the evaluation of $\hat{\mathcal{L}}(\xi_j, \theta_i; \mathbf{X})$, the quality of the estimation of the landscape of the loss function is then dependent on the specific set of data used to estimate it.

Optimization theory offers tools to deal with the trade-off between meaningfulness and efficiency. If we are willing to sacrifice part of the meaningfulness for efficiency, it is possible to define PROXY FUNCTIONS in substitution for a loss function. Ideally, a proxy function $\mathcal{L}' : \mathfrak{F} \to \mathbb{R}$ for the loss function $\mathcal{L} : \mathfrak{F} \to \mathbb{R}$ is a function that preserves the same extrema, such that $\operatorname{argmin}_{f \in \mathfrak{F}} \mathcal{L}' = \operatorname{argmin}_{f \in \mathfrak{F}} \mathcal{L}$. A proxy function $\mathcal{L}'$ may then be a simpler, better-behaved function that approximates $\mathcal{L}$ and whose optimization provides results identical or close to the optimization of $\mathcal{L}$.

2.2.3.5 Defining the optimization algorithm

Once we have defined a loss function, we can choose an optimization algorithm to solve the learning problem. In the ideal case, the optimal solution to a loss function $\mathcal{L}$ or its proxy $\mathcal{L}'$ is unique and can be found analytically. Convex functions, for instance, are a well-know family of functions that admits a unique solution to the minimization problem. More frequently, however, functions have a complex behaviour and a neat closed-form solution may not be computable. In these situations, it is possible to rely on optimization algorithms that, by exploring the space of solutions $\mathfrak{F}$, lead towards an optimal or sub-optimal solution. A family of well-known algorithms of this kind is the family of GRADIENT DESCENT algorithms. Assuming that a loss function is differentiable, the idea behind gradient descent is that, starting from a potential candidate solution f', it is possible to compute the gradient of the loss function $\nabla\mathcal{L}$ in f' and move the solution in the direction along which the loss function $\mathcal{L}$ decreases. Repeating this procedure allows us to keep improving the solution until an optimal or satisfactory solution is

reached. Several implementations and refinements of the basic gradient descent algorithm are available (MacKay, 2003).

2.2.3.6 Defining the use of data

Data play a central role in induction and in machine learning. Indeed, computational learning is essentially a data-driven process: data are used to set the parameters of the model to be learned, to evaluate the loss function and to decide how to proceed in the search for an optimal model. In other words, data are used to determine *how* we learn, *what* we learn and *how good* is what we learned.

In order to learn from a given set of data $\mathbf{X}$, data is required to satisfy a basic principle of consistency, guaranteeing that all the data represent the same process or the same entity. Statistically, the strongest way to assert this requirement translates in the following assumption:

Assumption (Independent and identically distributed (i.i.d.) data). *All the samples* $\mathbf{x}_i$ *are independently sampled from the same pdf* $p(X)$.

This assumption guarantees: (i) independence, meaning that all the data are generated independently without affecting each other; (ii) identical distribution, meaning that all the data are generated by the same process with no change happening over time in sampling. The assumption of i.i.d. data is a strong, but useful, assumption. Few real-world data sets perfectly comply with this ideal assumption, but they may be nonetheless modelled in this way provided that we are willing to accept such an approximation.

Given a set of i.i.d. data $\mathbf{X}$ from which to learn, it is possible to partition these data for the three different aims of learning good parameters, choosing good hyper-parameters and evaluating the degree of generalization achieved by the induction process:

- *Training data*: training data $\mathbf{X^{tr}}$ is defined as an exclusive subset of all the available data $\mathbf{X}$ that are used to learn the parameters θ_i. Using the training data we estimate $\hat{\mathcal{L}}\left(\theta_i; \xi_j, \mathbf{X^{tr}}\right)$, that is we compute the loss function with fixed hyper-parameters and fixed data. This allows us to evaluate the goodness of the current model f and to estimate the landscape of $\mathcal{L}$ simply as a function of the parameters θ_i. When using gradient descent, we can analyse this landscape and then find directions along which a modification of the parameters θ_i leads to improved solutions f. Thus, training data are used to determine *what* we learn.

- *Validation data*: validation data $\mathbf{X^{val}}$ is defined as an exclusive subset of all the available data $\mathbf{X}$ that are used to learn the hyper-parameters ξ_j. The hyper-parameters allow us to modify the behaviour of the meta-algorithm trying to learn a model. After optimizing the parameters θ_i on the training data, we can use the validation data to estimate $\hat{\mathcal{L}}\left(\xi_j; \theta_i, \mathbf{X^{val}}\right)$, that is to compute the loss function with fixed parameters and fixed data. If we have different sets of parameters θ_i optimized using different meta-models with hyper-parameters ξ_j, we can compare all of them and determine which one provides the

minimal loss $\hat{\mathcal{L}}\left(\xi_j; \theta_i, \mathbf{X}^{\mathbf{val}}\right)$. Notice that here we estimate the loss function not on the training data but on the new set of validation data because we do not want to select the hyper-parameters on the same set on which we optimized the parameters. This follows from the principle of generalization: we want a set of hyper-parameters defining a meta-model that is not optimized for a specific set of data $\mathbf{X}^{\mathbf{tr}}$, but which could explain any data set generated in the same way, as in the case of $\mathbf{X}^{\mathbf{val}}$. Thus, validation data are mainly used to determine *how* we learn. However, notice that, even if the hyper-parameters are not directly involved into the optimization process, they can sensibly affect the quality of the outcome of learning; poor choice of the hyper-parameters, such as an inadequate loss function or an improper setting of the optimization algorithm, may result in bad solutions, thus compromising the learning itself.

- *Test data*: test data $\mathbf{X}^{\mathbf{tst}}$ is defined as an exclusive subset of all the available data $\mathbf{X}$ that are used to evaluate the final goodness of the learned model f. After optimizing the parameters θ_i on the training data and choosing the hyper-parameters ξ_j on the validation data, we can use the test data to estimate $\hat{\mathcal{L}}\left(\xi_j, \theta_i, \mathbf{X}^{\mathbf{tst}}\right)$, that is to compute the loss function with all the arguments (hyper-parameters, parameters and data) fixed. Again, we use a specific data set for this estimation because of the principle of generalization: we want an estimate of the final goodness of the learned model f not on the data set $\mathbf{X}^{\mathbf{tr}}$ on which we optimized the parameters nor on the data set $\mathbf{X}^{\mathbf{val}}$ used to select the best hyper-parameters, but on a new unseen data set $\mathbf{X}^{\mathbf{tst}}$ which is generated in the same way as the previous one. Thus, test data are used to determine *how good* is what we learned.

2.2.3.7 Defining a performance measure

Finally, we want to define a quantitative performance measure to evaluate the quality of a learning machine algorithm, so that it may be compared to other algorithms. A learning machine is normally evaluated by how well the learned model fits the data. Its quality is rarely a binary matter of success or failure; more likely, it is an index of the degree to which the learned model properly describe the phenomenon under study.

A performance measure is then defined as a functional $\mathcal{P} : \mathfrak{F} \times \mathcal{X} \to \mathbb{R}$ which is evaluated using the learned model f and the test data $\mathbf{X}^{\mathbf{tst}}$.

It may be tempting to use the loss function chosen for optimization to compute the performance of a learning algorithm, but this may be not ideal. Indeed, the final measure of performance is meant to be meaningful, while the loss function is often transformed into a proxy function in order to trade meaning for efficiency. Differently from the loss function, the performance functional does not need to be iteratively evaluated and optimized, and therefore we do not have to worry about trading meaningfulness for efficiency. The performance functional can capture closely the idea of the goodness of a learned model, disregarding those requirements (such as differentiability) which are desirable for a loss function during learning.

Different types of performance measures may be defined according to the type of data we use, the type of model we are trying to learn, and the specific aim with which we may be concerned. For instance, specific metrics may be defined to measure the generalization performance of the learned model, the information content of the learned representations, the precision or

the sensitivity of the learned model (Murphy, 2012).

2.2.4 Taxonomy of machine learning problems

Having defined the terms within which we can design a machine learning algorithm `A`, we now move on to explore the different types of algorithms available in the machine learning literature. We will first offer an overview of the diversity of the problems in machine learning, and then we will provide a simple taxonomy to distinguish among machine learning problems of interest to this research.

2.2.4.1 Machine learning problems in literature

Machine learning literature abounds with several different types of learning problems. Beside some common and widely-accepted categories, several other types have been defined to handily cluster together problems and solutions having specific commonalities. As these new types have been defined in relation to particular needs, their boundaries are often fuzzy and they may easily overlap with other classes of learning problems.

In general, it is possible to identify some rigorous orthogonal conceptual dimensions to classify various types of machine learning problems. Because of the orthogonality of these categories, a single problem may be considered as belonging at the same time to different classes, even if this is rarely made explicit in the literature. We recall here the main classes of problems, referring the reader to the references for a detailed description of each one.

The most common dimension along which learning problems are classified is the *type of data* used for learning, thus giving rise to supervised learning (Bishop, 2007), unsupervised learning (Bishop, 2007), reinforcement learning (Sutton and Barto, 1998; Szepesvári, 2010), semi-supervised learning (Chapelle et al., 2006), self-taught learning (Raina et al., 2007), transductive learning (Arnold et al., 2007), self-supervised learning (Marblestone et al., 2016) and multi-view learning (Blum and Mitchell, 1998).

Another common dimension is the *aim of learning*, which induces the categories of feature learning (van Rooyen and Williamson, 2015), representation learning (Bengio et al., 2013), common feature learning (Argyriou et al., 2007), manifold learning (Lee and Verleysen, 2007), disentanglement learning (Desjardins et al., 2012), transfer learning (Pan and Yang, 2010), metric learning (Xing et al., 2003), semantic learning (Deerwester et al., 1990), causality learning Pearl (2009) and invariance learning (Larochelle et al., 2007).

A particularly important dimension is the one relative to the *theoretical justification* backing the machine algorithms, and which give rises to the categories of linear learning (Bishop, 2007), non-linear learning (Bishop, 2007), information-theoretic learning (Principe, 2010), energy-based learning (LeCun et al., 2006), variational learning (Bishop, 2007), Bayesian learning (MacKay, 2003), graph-based learning (Koller and Friedman, 2009), adversarial learning (Goodfellow et al., 2014), data distribution learning (Ngiam et al., 2011) and feature distribution learning (Ngiam et al., 2011).

A dimension which recently became popular is the *architecture* of the learning machine, specifically in reference to neural networks, that allows to distinguish between shallow learning

and deep learning (LeCun et al., 2015).

Sometimes, the *specific settings* of a learning problem may also spawn potential categorizations; for instance, it is possible to find references to life-long learning (Thrun and Pratt, 2012), learning to learn (Thrun and Pratt, 2012), context-sensitive learning (Turney, 2002), meta-learning (Brazdil et al., 2008), incremental or cumulative learning (Gepperth and Hammer, 2016), random learning (Saxe et al., 2011), cooperative learning (Panait and Luke, 2005), cost-sensitive learning (Elkan, 2001) and active learning (Cohn et al., 1994).

Finally, a high number of classes may be generated by considering the field of *application* of a learning machine, for instance vision learning or emotion learning.

In this dissertation, we will restrict our attention only to certain major learning problems, those which we review in the next section in relation to a taxonomy based on the definition of the space $\mathcal{Z}$ of the learned representations.

2.2.4.2 Taxonomy of machine learning problems by space of representations

A basic classification of machine learning problems is based on the definition of the space of the representations $\mathcal{Z}$.

In the standard machine learning setting, the space of the original representations $\mathcal{X}$ is fixed and defined a priori. Because of the assumption of i.i.d. data, we are given a set of homogeneous[3] data samples $\mathbf{x}_i$ all coming from the same space $\mathcal{X}$.

On the other hand, the space of the learned representations $\mathcal{Z}$ is similarly homogeneous, but its nature and shape may or may not be known a priori. It is then possible to distinguish three main classes of machine learning problems.

Supervised learning. Learning problems where the space of the learned representations $\mathcal{Z}$ is defined are called *supervised learning problems*. In these cases, the nature of the space of the learned representations $\mathcal{Z}$ (e.g., continuous, discrete, categorical) and its cardinality are known.

Practically, information about the nature and the cardinality of the space of learned representations is provided along the data samples $\mathbf{x}_i$, often in the form of LABELS $\mathbf{y}_i$ associated with each data sample. Labels are then a specific type of learned representations which encode human-provided knowledge in a highly abstract form. Thus $\mathbf{Z} = \mathbf{Y}$; given their peculiarity, we will use the notation $\mathbf{Y}$ to distinguish labels from other types of generic representations $\mathbf{Z}$.

The objective of supervised learning is to discover a model $f \in \mathfrak{F}$ that successfully maps the original representations $\mathbf{X}$ to the labels $\mathbf{Y}$ or, statistically, to discover the distribution $p(X, Y)$ or $p(Y|X)$ that explains the data and the labels. Supervised learning may be seen as the area of research concerned with the *problem of modelling a functional relationship*, that is finding a deterministic or stochastic law between a set of data and a set of labels that are assumed to depend (logically, stochastically, or causally) on the data.

Knowledge of the space of the learned representations $\mathcal{Z}$ provides relevant information on the space of learnable morphisms $\mathfrak{F}$, too. Indeed, given the knowledge of the nature and the cardinality of the space of the learned representations $\mathcal{Z}$, we have perfect knowledge not only

[3] The case of heterogeneous data points $\mathbf{x}_i$ coming from different spaces is usually not considered, and it would form a class of learning problems of its own.

of the domain of the map f, but also of its co-domain. If f is a single function, then it is possible to easily restrict the space $\mathfrak{F}$ to the family of functions $\mathcal{X} \to \mathcal{Y}$ and sensibly simplify the problem of a finding a proper model; if f is given by the composition of multiple functions $f^{(n)} \circ \ldots \circ f'' \circ f'$, it is still possible to set immediately the domain of the first function f' and the co-domain of the last function $f^{(n)}$.

According to the nature and the cardinality of $\mathcal{Y}$, it is possible to identify different sub-types of supervised learning problems:

- *Classification:* supervised learning problems where the domain of the labels is a one-dimensional finite discrete set are defined as classification problems, $\mathcal{Y} \subseteq \mathbb{N}$. Labels $\mathbf{Y}$ represent unique and disjoint categories that partition the set of observations $\mathbf{X}$. Every observation belongs unambiguously to one out of $C \in \mathbb{N}_{>0}$ categories and the aim of a learning machine is to discover the model that can extract the high-level knowledge contained in the labels from the finite, low-level and noisy information in the data.

  Well-known, concrete examples of learning machines for classification are Support Vector Machines (SVM) (Cortes and Vapnik, 1995), perceptrons (Rosenblatt, 1958) or K-Nearest Neighbours (KNN) (Fix and Hodges Jr, 1951).

- *Regression:* supervised learning problems where the domain of the labels is a one-dimensional continuous set are defined as regression problems, $\mathcal{Y} \subseteq \mathbb{R}$. Labels $\mathbf{Y}$ represent the output of a complex function of the observations $\mathbf{X}$. The aim of a learning machine is to find out the model that can extract the high-level knowledge contained in the outputs from the finite, low-level and noisy information in the data.

  A well-known, concrete example of learning machines for regression is the Ordinary Least Squared (OLS) regressor (Bishop, 2007).

- *Multi-task learning:* classification or regression problems where the domain of the labels is multi-dimensional are defined as multi-task learning problems, $\mathcal{Y} \subseteq \mathbb{N}^L$ or $\mathcal{Y} \subseteq \mathbb{R}^L$. Labels are either collections of classes relative to different categorizations or collections of outputs of different complex functions. The aim of a learning machine is to discover a multi-valued map $f : \mathcal{X} \to \mathbb{N}^L$ or $f : \mathcal{X} \to \mathbb{R}^L$ that can solve the different classification problems or the different regression problems. Most likely, solving a multi-task learning problem is more than the plain sum of the solution of multiple independent classification or regression tasks; multi-task learning aims at exploiting the commonalities between different problems and sharing information across the diverse tasks.

  A well-known, concrete example of learning machines for multi-task learning is the Multi-Task Learning (MTL) back-prop network (Bishop, 2007).

Unsupervised learning. Learning problems where the space of the learned representations $\mathcal{Z}$ is not defined a priori along with the input data are defined as *unsupervised learning problems*. In these cases, the nature and the cardinality of the space of the learned representations $\mathcal{Z}$ are choices left to the designer of the machine learning algorithm. Notice that ignorance about the nature and the cardinality of the space $\mathcal{Z}$ of the learned representations may severely affect the possibility of effectively restricting the research space $\mathfrak{F}$.

Contrasted with supervised learning where labels are provided for learning a relationship between the data and the labels, unsupervised learning cannot rely on any external semantics in the form of labels. Thus, in order to direct learning, unsupervised learning must rely on the specification of assumptions and constraints. These assumptions and constraints are used to translate the very understanding of the problem of learning and encode the intuition about what types of mappings are useful or meaningful. Differently from supervised learning, unsupervised learning can be seen as the area of research concerned with the *problem of modelling the data*. Modelling the data through the learning of new representations may be an aim in itself, but most commonly it is instrumental for further steps of learning. Engineering, especially signal processing, provides many examples of hand-coded transformations that generate better representations of a signal for further elaboration (e.g., amplification, noise filtering, Fourier transform); similarly, unsupervised learning aims at learning useful transformation in an automatic way.

Thus, given the data $\mathbf{X}$, the tacit aim of unsupervised learning may often be to generate new representations $\mathbf{Z}$ that simplify the ensuing problem of learning meaningful relationships with respect to a set of labels $\mathbf{Y}$. This requires mapping the original representations, which may be described by a complex and ill-behaved conditional distribution $p(Y|X)$, onto new representations that can be described by a better-behaved and easier-to-learn conditional distribution $p(Y|Z)$.

According to the aim of unsupervised learning and the shape of $\mathcal{Z}$, it is possible to identify different sub-types of unsupervised learning problems.

- *Clustering:* unsupervised learning problems where the aim of learning is to group together observations that happens to be similar are defined as clustering problems. Similarity between observations is defined according to a chosen metric. In standard (or hard) clustering, the domain of the learned representations is taken to be a one-dimensional finite discrete set, $\mathcal{Z} \subseteq \mathbb{N}$. Original representations that, by virtue of their similarity, are mapped onto the same learned representation are taken to belong to the same cluster. Ideally, clusters provide a form of high-level information by defining meaningful groupings of the data and by collecting together samples with similar characteristics.

  Clustering can be seen as the unsupervised counterpart of classification; in clustering observations are categorized without having explicit outside information on how to define the classes. In order to perform clustering, it is then necessary to determine two elements: (i) the number of clusters to consider; (ii) the metric that will be used to evaluate the similarity among observations. These choices are specific to each learning machine.

  A well-known, concrete example of a learning machine for clustering is K-means (Forgy, 1965).

- *Soft clustering:* unsupervised clustering problems in which an observation can belong to multiple clusters are defined as soft clustering problems. In soft clustering, the domain of the learned representations is taken to be a multi-dimensional set, $\mathcal{Z} \subseteq \mathbb{N}^L$ or $\mathcal{Z} \subseteq \mathbb{R}^L$. Observations are not bound to belong to a single category anymore, but they can belong to different categories with different degrees of association. Similarly to hard clustering,

soft clustering is expected to provide a form of high-level and useful information by decomposing the original representations into non-exclusive clusters that describe aspects of the observations or encode different characteristics of the data.

Soft clustering can be seen as the unsupervised counterpart of multi-task classification; in soft clustering observations are categorized in multiple classes without having explicit outside information on how to define the classes. In order to perform clustering, it is then necessary to determine two elements: (i) the number of clusters to consider; (ii) a rule defining how observations can belong to multiple classes; (iii) the metric that will be used to evaluate the similarity among observations. These choices are specific to each learning machine.

A well-known, concrete example of a learning machine for clustering is Gaussian Mixture Models (GMM) (MacKay, 2003).

- *Representation learning:* unsupervised learning problems where the aim of learning is to generate better representations of the original observations are defined as representation learning problems. In representation learning, the domain of the learned representations is a generic multi-dimensional continuous set, $\mathcal{Z} \subseteq \mathbb{R}^L$. Representation learning intuitively defines better representations as representations with implicit properties such as *denoising* (representations where noise has been filtered out), *robustness* (representations insensitive to random fluctuations), *simplicity* (representations that behave in a regular way), *disentanglement* (representations separating different information components) and *informativeness* (representations having a high informational content) (Bengio et al., 2013).

  As it may be evinced by the vague definitions of what makes a representation better than another, representation learning encompasses a very wide class of algorithms, and this term may be used as an umbrella term to embrace any machine learning algorithm that produces a learned representation without an explicit external aim; specific forms of unsupervised learning, such as clustering, dimensionality reduction, or distribution learning may all be labelled, in generic terms, as forms of representation learning.

  A well-known, concrete example of a learning machine for representation learning is Auto-Encoders (AE) (Bengio et al., 2013).

- *(Unsupervised) Dimensionality Reduction:* unsupervised representation learning problems where the aim of learning is to generate representations with a dimensionality lower than that of the original representations are defined as unsupervised dimensionality reduction problems. In dimensionality reduction, the domain of the original representations is $\mathcal{X} \subseteq \mathbb{R}^M$, while the domain of the learned representations is $\mathcal{Z} \subseteq \mathbb{R}^L$, with the strict constraint $L < M$. The objective of dimensionality reduction is to generate useful and high-level representations by producing compressed representations of the original data. A lower-dimensional representation in $\mathcal{Z}$ is expected to be useful on the ground that it provides a more essential, ideally noiseless, representation of the original data.

  Well-known, concrete examples of learning machines for dimensionality reduction are

Principal Components Analysis (PCA) (Pearson, 1901) or Isomap (Tenenbaum et al., 2000).

- *Data distribution learning:* unsupervised learning problems where the aim of learning is deriving the original distribution $p(X)$ that underlies the generation of the data are defined as data distribution learning (DDL) problems. Through DDL, new learned representations $\mathbf{Z}$ are generated with a distribution $p(Z)$ that tries to match as closely as possible the actual distribution $p(X)$. Ideally, the learned representations $\mathbf{Z}$ will be high-level and useful representations, under the assumption that $\mathbf{Z}$ will encode the factors that explain the original data $\mathbf{X}$ and that such knowledge may be successfully exploited for learning a functional relationship.

  Well-known, concrete examples of DDL algorithms (Ngiam et al., 2011) include denoising auto-encoders (DAE) (Vincent et al., 2008a), restricted Boltzmann machines (RBM) (Hinton et al., 2006) and independent component analysis (ICA) (Bell and Sejnowski, 1997).

- *Feature distribution learning:* unsupervised learning problems where the aim of learning is deriving a distribution $p(Z)$ with a set of desirable properties are defined FDL problems. In contrast to DDL, FDL algorithms overlook the problem of estimating the distribution $p(X)$ and focus instead on shaping the learned distribution $p(Z)$ according to chosen criteria. FDL algorithms are usually instrumental, in that they are meant to shape a pdf $p(Z)$ which has useful properties for some ensuing task.

  A well-known, concrete example of FDL is SF (Ngiam et al., 2011).

Semi-supervised learning. Learning problems where the space of the learned representations $\mathcal{Z}$ is defined a priori only for a subset of the given data are defined as *semi-supervised learning problems*. In this case, information on the nature and shape of $\mathcal{Z}$ is provided only for some samples, but, relying on the i.i.d. assumption, it is normally possible to extend this information to the unsupervised samples too. This situation is common when, for practical reasons (e.g., cost of labelling), it is impossible to provide learned representations $\mathbf{Z}$ or labels $\mathbf{Y}$ for all the data samples $\mathbf{X}$.

The N available data samples $\mathbf{X}$ can be partitioned into a supervised set $\mathbf{X^{sup}}$ containing $N^{sup} \in \mathbb{N}_{>0}$ observations and an unsupervised set $\mathbf{X^{unsup}}$ containing $N^{unsup} \in \mathbb{N}_{>0}$ observations. As in the case of supervised learning, the supervised set $\mathbf{X^{sup}}$ is associated with its learned representations $\mathbf{Z}$, usually in the form of labels, so that $\mathbf{Z} = \mathbf{Y}$.

The overall aim of semi-supervised learning is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$, with the requirement of exploiting all the available data, including the unlabelled data $\mathbf{X^{unsup}}$. The requirement of exploiting the information conveyed by the unlabelled data stems from the principle of preservation of data. Thus, semi-supervised learning tries to harness the information provided by the large amount of unlabelled data in order to improve supervised learning.

Depending on to the available labelled data $\mathbf{X^{sup}}$ and unlabelled data $\mathbf{X^{unsup}}$, it is possible to identify the following form of semi-supervised learning:

(Inductive) Learning $f : \mathcal{X} \to \mathcal{Z}$	**Supervised Learning:** $\{\mathbf{X}, \mathbf{Y}\}$ $f : \mathcal{X} \to \mathcal{Y}$	*Classification* $\mathcal{Y} \subseteq \mathbb{N}$ *Regression* $\mathcal{Y} \subseteq \mathbb{R}$ *Multi-task Learning* $\mathcal{Y} \subseteq \mathbb{N}^L$ or $\mathcal{Y} \subseteq \mathbb{R}^L$
	Unsupervised Learning: $\{\mathbf{X}\}$ $f : \mathcal{X} \to \mathcal{Z}$	*Clustering* $\mathcal{Z} \subseteq \mathbb{N}$ *Soft Clustering* $\mathcal{Z} \subseteq \mathbb{N}^L$ or $\mathcal{Z} \subseteq \mathbb{R}^L$ *Representation Learning* $\mathcal{Z} \subseteq \mathbb{R}^L$ *Dimensionality Reduction* $\mathcal{Z} \subseteq \mathbb{R}^L$ s.t. $M < L$ *Data Distribution Learning* $\mathcal{Z} \subseteq \mathbb{R}^L$ s.t. $p(Z) \approx p(X)$ *Feature Distribution Learning* $\mathcal{Z} \subseteq \mathbb{R}^L$ s.t. $p(Z)$ has specific properties
	Semi-supervised Learning: $\{\mathbf{X^{sup}}, \mathbf{Y}, \mathbf{X^{unsup}}\}$ $f : \mathcal{X} \to \mathcal{Z} \to \mathcal{Y}$	*Semi-supervised Learning* $\mathcal{Z} \subseteq \mathbb{R}^L$ and $\mathcal{Y} \subseteq \mathbb{N}$

Table 2.3: Taxonomy of learning problems.

- *(Standard) Semi-supervised learning:* semi-supervised learning works with i.i.d. labelled data $\mathbf{X^{sup}}$ and unlabelled data $\mathbf{X^{unsup}}$ coming from a same domain and the same distribution. The aim is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$, usually via a first unsupervised mapping $f^{unsup} : \mathcal{X} \to \mathcal{Z}$ exploiting labelled and unlabelled data, and a second supervised mapping $f^{sup} : \mathcal{Z} \to \mathcal{Y}$ relying only on the labelled data. As the name may suggest, semi-supervised learning can be seen as a combination of unsupervised learning and supervised learning, and the overall learned mapping f can be seen as a composition of an unsupervised and a supervised mapping: $f = f^{sup} \circ f^{unsup}$.

Table 2.3 summarizes the taxonomy of the different forms of learning problems that we considered.

After having presented how machine learning formalizes different types of learning problems, in the next two sections we will narrow our focus to two very specific forms of learning: FDL and CSA.

2.3 Feature Distribution Learning

Section 2.2.4 presented and categorized many problems tackled by machine learning. After this overview, the present section concentrates on a recent form of unsupervised learning that has

shown good promise: FDL. Section 2.3.1 starts by describing in detail the standard approach to unsupervised learning called data distribution learning (DDL). Section 2.3.2 presents FDL by contrasting it with DDL. Section 2.3.3 temporarily diverts the discussion by introducing a popular technique applied to unsupervised learning, that is, sparsity learning. Section 2.3.4 brings us back to the topic of FDL and combines it with sparsity learning by introducing the first prototypical FDL algorithm, that is, SF. Finally, Section 2.3.5 reviews the current state of the art on the research and application of FDL and SF.

2.3.1 Data distribution learning

Most representation learning algorithms may be described as DDL algorithms, that is, algorithms that try to learn new representations which can better model the underlying probability distribution that generated the data. Estimating the original pdf $p(X)$ from sample data $\mathbf{X}$ in high dimensions is very challenging in itself and requires reliable and efficient statistically-grounded algorithms. Moreover, the data $\mathbf{X}$ cannot usually be considered noiseless; part of the variability observed in the data is actually due to the variance in the process, but part of it may be due to other random processes that superimpose noise on the data. Formally, DDL algorithms often make the following assumption:

Assumption (Noisy Sampling). *Samples are first generated by a stochastic process with pdf $p(X^*)$, which we refer to as the* true *pdf. Samples from $p(X^*)$ are corrupted by various forms of noise, such that the noisy samples $\mathbf{x}_i$ follow a new noisy pdf $p(X)$.*

This assumption postulates the existence of an underlying *true*[4] *pdf* $p(X^*)$ that generated the data. Unfortunately, clean samples from $p(X^*)$ cannot be accessed in reality and the original samples $\mathbf{x}_i$ that we receive are corrupted by various forms of noise. Because of this injection of noise, the original representations $\mathbf{X}$ we end up working with follow a new distribution $p(X)$.

Recovering the true distribution $p(X^*)$ is important in order to have a better description of the phenomenon under study. In particular, suppose we were to be provided with labels $\mathbf{Y}$ associated with the data $\mathbf{X}$. It is reasonable to make the following assumptions:

Assumption (Correlation between Labels and True Distribution). *Noiseless samples generated by the true pdf $p(X^*)$ have a stronger correlation to the labels y_i than the original noisy samples $\mathbf{x}_i$*

This assumption reasonably states that the correlation between the samples from the pdf $p(X^*)$ and the labels is stronger than the correlation between the noisy samples from the pdf $p(X)$ and the labels. This follows from the definition of noise, which is essentially unrelated to any meaningful task at hand. If this assumption holds, it makes sense to learn new representations with a pdf that approximates the true distribution, $p(Z) \approx p(X^*)$. In this way, supervised learning with respect to the labels $\mathbf{Y}$ would be eased because of the higher correlation between $p(Z)$ or $p(X^*)$ and $\mathbf{Y}$, thus making it easier to learn $p(Y|Z)$ or $p(Z,Y)$ than $p(Y|X)$ or $p(X,Y)$.

[4] The term *true* is used with no absolute epistemological meaning, but just in reference to our relative interest; the true distribution is the noiseless distribution that describes the phenomenon we are interested in studying.

2.3.2 Feature distribution learning

DDL is a very widespread approach to unsupervised learning, but learning may be directed in alternative ways. Ngiam et al. (2011) proposed a novel unsupervised learning framework for generating sparse representations based on the intuition of learning new representations of the data by focusing only on the output, instead of the input, of the representation learning process.

In contrast to DDL, Ngiam et al. (2011) proposed the possibility of developing FDL algorithms, that is, algorithms which try to learn a new representation having desirable properties, without taking into account the problem of modelling the distribution of the data. The properties to be learned are chosen with respect to other ensuing tasks, such as classification. In this case, properties like sparsity or smoothness are well known to be useful when modelling $p(Y|Z)$ or $p(Z,Y)$ and, therefore, they could be directly learned by a FDL algorithm.

2.3.3 Learning sparsity

One common algorithmic choice hard-wired in several unsupervised learning algorithms is *sparsity* (Bengio et al., 2013). Sparse representation learning aims at learning a mapping that produces a new representation $\mathbf{Z}$ where few of the components are active while all the others are reduced to zero.

Sparsity. Given a generic vector $\mathbf{z}_i$ in an L-dimensional space, we say that $\mathbf{z}_i$ is sparse if a small number of components of the vector accounts for most of the energy[5] of the vector (Hurley and Rickard, 2009).

Practically, we say that:

- A vector $\mathbf{z}_i$ is sparse if $l \ll L$ components of the vector $\mathbf{z}_i$ are active, that is, they have a value different from zero, while the remaining $L - l$ components are inactive, that is, they have the value zero.
- A vector $\mathbf{z}_i$ is ϵ-sparse if $l \ll L$ components of the vector $\mathbf{z}_i$ are active, that is, they have a value greater than ϵ, while the remaining $L - l$ components are inactive, that is, they have a value less than ϵ.
- A vector $\mathbf{z}_i$ is k-sparse if exactly k components are active (Makhzani and Frey, 2013).

By analogy, we may define sparsity for matrices (with reference to their components) and for random variables (with reference to their realizations). When dealing with a matrix $\mathbf{Z}$ made up of N samples in L dimensions, it is possible to identify different types of sparsity:

- *Population sparsity* is the sparsity computed with respect to the samples. $\mathbf{Z}$ satisfies population sparsity if, for all $0 < i \leq N$, $\mathbf{z}_i$ is sparse. Population sparsity forces each sample $\mathbf{z}_i$ to be described by only a few features. Two different examples of matrices satisfying population sparsity are illustrated in Figures 2.1(a) and 2.1(b).

[5] We refer here to the meaning of energy from signal processing, that is: $energy(\mathbf{z}_i) = \sum_{j=1}^{L} |z_{i,j}|^2$.

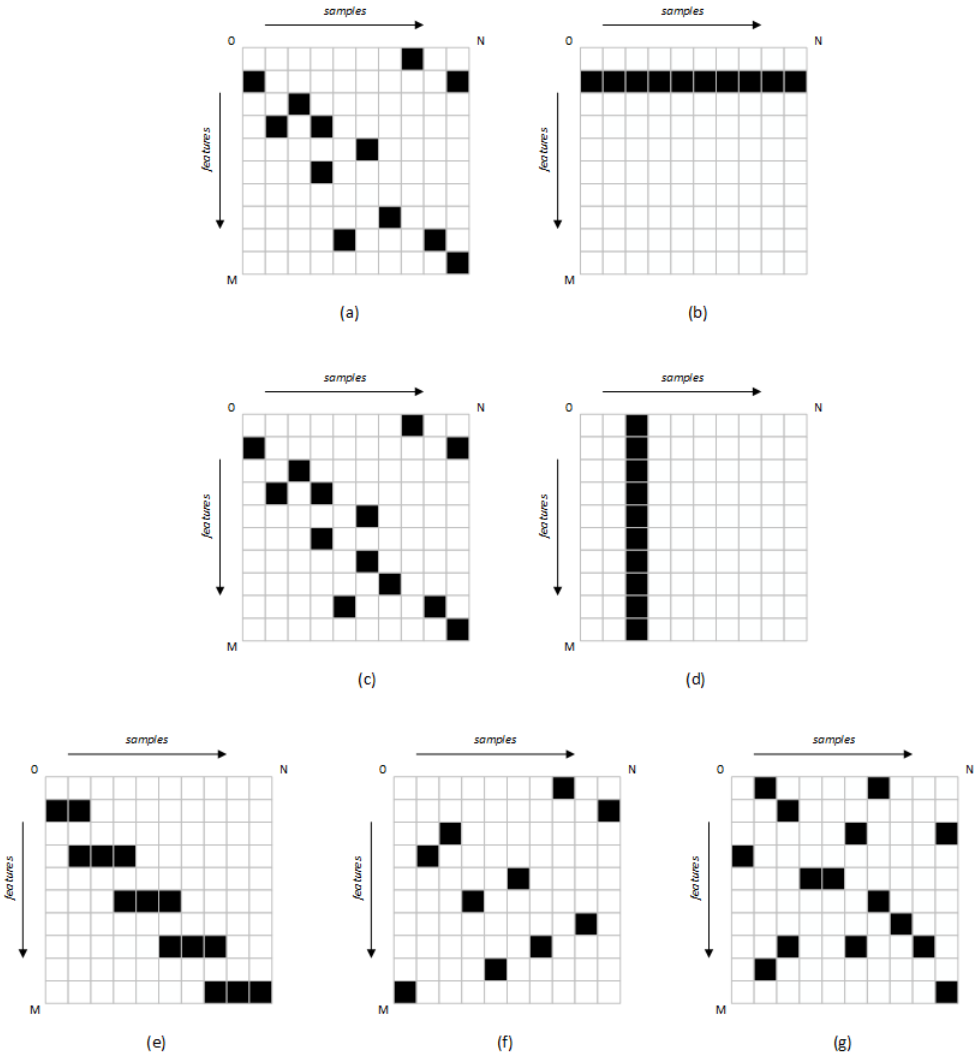

Figure 2.1: Example of sparse matrices illustrating different properties of sparsity (population sparsity, lifetime sparsity, high dispersal).
Each matrix is composed by M features on the rows and N samples on the columns. A white square means that the element $z_{i,j}$ of the matrix is inactive, while a black square means that the element $z_{i,j}$ of the matrix is active. The computation of the values of the properties of population sparsity, lifetime sparsity, high dispersal is listed in Table 2.4.

- *Lifetime sparsity* or *selectivity* (Goh et al., 2012) is the sparsity computed with respect to the features. $\mathbf{Z}$ satisfies lifetime sparsity if, for all $0 < j \leq L$, $\mathbf{z}_{\cdot,j}$ is sparse. Lifetime sparsity forces each feature $\mathbf{z}_i$ to be descriptive of only a few samples. Two different examples of matrices satisfying lifetime sparsity are illustrated in Figures 2.1(c) and 2.1(d).

- *High dispersal* is a form of constrained sparsity with respect to the features. $\mathbf{Z}$ satisfies high dispersal if, for all $0 < j', j'' \leq L$, $\mathbf{z}_{\cdot,j'}$ and $\mathbf{z}_{\cdot,j''}$ have approximately the same activation. Practically high dispersal may be enforced by imposing a low variance on every sparse feature, $Var[Z_{\cdot,j}] < c$, for an arbitrarily small constant c. High dispersal causes each feature to be approximately equally active. A negative example of a matrix with a degenerate pattern that does not satisfy high dispersal is illustrated in Figure 2.1(e).

Assuming that each sample is described by at least one active feature, enforcing the three properties described above leads to the learning of distributed sparse patterns and prevents the learning of degenerate sparse representations. Examples of matrices satisfying all the properties of population sparsity, lifetime sparsity and high dispersal are provided in Figures 2.1(f) (which shows a case having the best possible values of population sparsity, lifetime sparsity and high dispersal) and 2.1(g). Table 2.4 reports the explicit values of population sparsity, lifetime sparsity and high dispersal for all the examples in Figure 2.1.

Measuring sparsity. The definition given above highlights the fact that sparsity can hardly be conceived of as a strict binary property of a vector $\mathbf{z}_i$ or of a matrix $\mathbf{Z}$ intrinsic to their representations. Indeed, determining whether a vector $\mathbf{z}_i$ is sparse or not depends upon the choice of a specific sparsity parameter: m (in the case of simple sparsity), ϵ (in the case of ϵ-sparsity), k (in the case of k-sparsity). It is useful, however, to have functions that allow us to measure the degree of sparsity of a vector independently from any sparsity parameter. A parameter-free measure would save us the problem of selecting the proper value for the sparsity parameter and would allow us to minimize or maximize sparsity just by optimizing a single measure.

Several parameter-free measures of sparsity have been proposed in the literature (Hurley and Rickard, 2009). Here, we recall the popular norm-based measures of sparsity, which are relevant to SF and other SF-based algorithms. Norm-based measures of sparsity rely on the computation of an ℓ_p-norm of the vector $\mathbf{z}_i$.

The most direct and intuitive measure of sparsity is the ℓ_0-"norm". Given the generic vector $\mathbf{z}_i$, its ℓ_0-"norm" is defined as: $\ell_0(\mathbf{z}_i) = \sum_{j=1}^{L} \mathbf{1}_{\mathbb{R}\setminus\{0\}}(z_{i,j})$, where $\mathbf{1}_A(z)$ is the indicator function, returning 1 if $z \in A$ or returning 0 otherwise. In other words, the ℓ_0-"norm" simply returns the count of components in the vector $\mathbf{z}_i$ having value different from 0. Despite its simplicity, the ℓ_0-"norm" has two significant limitations. First, $\ell_0(\mathbf{z}_i)$ is inappropriate if, for computational reasons, the value of the features can never be perfectly zero. Second, $\ell_0(\mathbf{z}_i)$ is a non-smooth function with respect to $\mathbf{z}_i$, characterized by plateaus with a null derivative and steps with an infinite derivative. These limitations make it problematic to adopt $\ell_0(\mathbf{z}_i)$, especially when relying on derivative-based techniques for optimization.

Figure	Population Sparsity	Lifetime Sparsity	High Dispersal
2.1(a)	✓: $\{1,2\}$-sparse	✓: $\{0,2\}$-sparse	✓: $\hat{Var}[Z_{\cdot,j}] = 0.36$ $\hat{E}[Z_{\cdot,j}] = \frac{12}{10}$, $\hat{E}\left[Z^2_{\cdot,j}\right] = \frac{18}{10}$
2.1(b)	✓: 1-sparse	×: $\{0,N\}$-sparse	×: $\hat{Var}[Z_{\cdot,j}] = 9$ $\hat{E}[Z_{\cdot,j}] = 1$, $\hat{E}\left[Z^2_{\cdot,j}\right] = 10$
2.1(c)	✓: $\{1,2\}$-sparse	✓: $\{1,2\}$-sparse	✓: $\hat{Var}[Z_{\cdot,j}] = 0.21$ $\hat{E}[Z_{\cdot,j}] = \frac{13}{10}$, $\hat{E}\left[Z^2_{\cdot,j}\right] = \frac{19}{10}$
2.1(d)	×: $\{0,L\}$-sparse	✓: 1-sparse	✓: $\hat{Var}[Z_{\cdot,j}] = 0$ $\hat{E}[Z_{\cdot,j}] = 1$, $\hat{E}\left[Z^2_{\cdot,j}\right] = 1$
2.1(e)	✓: $\{1,2\}$-sparse	✓: $\{0,3\}$-sparse	×: $\hat{Var}[Z_{\cdot,j}] = 2.04$ $\hat{E}[Z_{\cdot,j}] = \frac{14}{10}$, $\hat{E}\left[Z^2_{\cdot,j}\right] = 4$
2.1(f)	✓: 1-sparse	✓: 1-sparse	✓: $\hat{Var}[Z_{\cdot,j}] = 0$ $\hat{E}[Z_{\cdot,j}] = 1$, $\hat{E}\left[Z^2_{\cdot,j}\right] = 1$
2.1(g)	✓: $\{1,2\}$-sparse	✓: $\{1,3\}$-sparse	✓: $\hat{Var}[Z_{\cdot,j}] = 0.45$ $\hat{E}[Z_{\cdot,j}] = \frac{15}{10}$, $\hat{E}\left[Z^2_{\cdot,j}\right] = \frac{27}{10}$

Table 2.4: Computed values of population sparsity, lifetime sparsity and high dispersal for the matrices in Figure 2.1.
Given $N = 10$ and $L = 10$, a sample $\mathbf{z}_i$ is assumed to satisfy population sparsity if the number of active features is less than or equal to 3; a feature $\mathbf{z}_{\cdot,j}$ is assumed to satisfy lifetime sparsity if it is active for 3 or fewer samples; a feature $\mathbf{z}_{\cdot,j}$ is assumed to satisfy high dispersal if $Var[Z_{\cdot,j}] < 1$.

A simple workaround to solve the first problem is to adopt on a $\ell_{0,\epsilon}$-"norm". Given the generic vector $\mathbf{z}_i$, its $\ell_{0,\epsilon}$-"norm" is defined as: $\ell_{0,\epsilon}(\mathbf{z}_i) = \sum_{j=1}^{L} \mathbf{1}_{\mathbb{R}\setminus[0,\epsilon]}(|z_{i,j}|)$, where $[0,\epsilon]$ is a closed set on $\mathbb{R}$. In this case, any component in the vector $\mathbf{z}_i$ having a value between 0 and ϵ is treated as a zero, and the $\ell_{0,\epsilon}$-"norm" returns the count of components greater than ϵ. Unfortunately, this solution is not parameter-free, as it requires the definition of the parameter ϵ, and does not address the problem of derivability, being again non-smooth in the neighbourhood of ϵ.

One of the most commonly adopted measures of sparsity is the ℓ_1-norm, often referred to as *activation*. Given the generic vector $\mathbf{z}_i$, its ℓ_1-norm is defined as: $\ell_1(\mathbf{z}_i) = \sum_{j=1}^{L} |z_{i,j}|$. The ℓ_1-norm simply returns the sum of the absolute value of the features of the vector $\mathbf{z}_i$. The ℓ_1-norm thus approximates the ℓ_0-"norm". While the ℓ_0-"norm" computes the number of active features of $\mathbf{z}_i$, the ℓ_1-norm computes the magnitude of the active components. In the case in which $\mathbf{z}_i$ is a binary vector, then $\ell_0(\mathbf{z}_i) = \ell_1(\mathbf{z}_i)$. When $\mathbf{z}_i$ is not a binary vector, then the ℓ_1-norm can be seen as a relaxation of the ℓ_0-"norm" (Elad, 2010). The ℓ_1-norm is a smooth function with continuous derivatives except in the origin, and it can therefore be used for optimization. Maximization of sparsity is then expressed as the minimization of the ℓ_1-norm (or as the maximization of the negative ℓ_1-norm).

Relevance of sparsity. The adoption of sparsity relies both on theoretical justifications and on biological analogies. From a formal perspective, sparse representations provide a good trade-off between robustness, maximized by one-hot representations, and representative power, maximized by dense representations (Bengio, 2009); they may be insensitive to irrelevant perturbations of the inputs (Bengio et al., 2013); they may help dealing with explaining-away phenomena (Bengio et al., 2013); they may improve pattern discrimination (Goh et al., 2014); they may help tackling the curse of high dimensionality (Ganguli and Sompolinsky, 2012); they allow high levels of compression within the compressed sensing framework (Candes et al., 2006); and the adoption of sparsity guarantees optimal information transmission by limiting the degeneracy between the input domain and the output domain, thus maximizing the mutual information between input and output (Kouh, 2017). From a physical and biological perspective, sparse representations are energy efficient; they are parsimonious and can enhance storage capacity (Rolls and Treves, 1990); several physical phenomena may be encoded with a sparse representation in a proper domain (Ganguli and Sompolinsky, 2012); the visual cortex (Olshausen and Field, 1997) and the brain cortex in general (Földiák and Young, 1995) seem to rely largely on sparse codes; and sparse codes make it possible to control the variability and the overlap of input stimuli to the brain (Babadi and Sompolinsky, 2014). Beyond these rationales, the usefulness of sparsity has been confirmed in numerous practical machine learning implementations (see, for instance, Bengio et al., 2013; Coates and Ng, 2011; Ranzato et al., 2006) and several different algorithms have been developed or have been adapted to learn sparse representations (Zhang et al., 2015).

2.3.4 Sparse filtering

In the previous sections we reviewed the ideas of feature distribution learning and sparsity learning, and now we will examine how they can be integrated in a concrete algorithm.

Working within the FDL framework, Ngiam et al. (2011) defined the SF algorithm as an instantiation of the FDL paradigm. SF is an unsupervised algorithm that ignores the problem of learning the distribution of the data and instead focuses only on the optimization of the sparsity of the learned representations. SF learns a maximally sparse representation via a mapping $\texttt{SF} : \mathbb{R}^M \to \mathbb{R}^L$ defined by the following transformation:

$$\mathbf{Z} = \ell_{2,col}\left(\ell_{2,row}\left(|\mathbf{W}\mathbf{X}|\right)\right),$$

where $\mathbf{W} \in \mathbb{R}^{L \times M}$ is a weight matrix, $|\cdot|$ is the element-wise absolute-value function, $\ell_{2,row}(\cdot)$ is the ℓ_2-normalization along the rows $\ell_{2,row}(\mathbf{X}) = \ell_{2,row}(x_{i,j}) = \frac{x_{i,j}}{\sqrt{\sum_{i=1}^{N}(x_{i,j})^2}}$, and $\ell_{2,col}(\cdot)$ is the ℓ_2-normalization along the columns $\ell_{2,col}(\mathbf{X}) = \ell_{2,col}(x_{i,j}) = \frac{x_{i,j}}{\sqrt{\sum_{j=1}^{M}(x_{i,j})^2}}$. During learning, the weight matrix $\mathbf{W}$ is trained by gradient descent in order to minimize the loss function $\mathcal{L} = \min_{\mathbf{Z} \in \mathbb{R}^{L \times N}} \sum_{i=1}^{N} \sum_{j=1}^{L} z_{i,j}$. A deeper theoretical analysis of this algorithm is the topic of Chapter 3.

SF proved to be an excellent algorithm for unsupervised learning: it scales very well with the dimensionality of the input; it is easy to implement; it was shown to achieve state-of-the-art performance; and it is extremely simple to tune, since it is "essentially hyperparameter-free", having only a single hyper-parameter to select (Ngiam et al., 2011). Excluding the definition of the termination conditions for the training, the SF algorithm has the single hyper-parameter L, that is, the dimensionality of the learned space, and this allows for an effective exploration of the space of the hyper-parameters even with a limited amount of computational power by preventing the exponential explosion of the possible combinations of hyper-parameters.

2.3.5 Overview of the state of the art

As SF is so far the only acknowledged FDL algorithm available, most of the state of the art of the research in the field of FDL corresponds to the state of the art of the research on SF.

Thanks to its simplicity and its performance, the SF algorithm has been studied and adopted by several researchers. In the original paper, Ngiam et al. (2011) showed that SF was able to provide state-of-the-art performance on image recognition and phone classification. Thaler (see Goodfellow et al., 2013) and Romaszko (2013) used SF in their machine learning systems while taking part into the Kaggle Black Box Learning Challenge[6] organized during the 2013 International Conference on Machine Learning, achieving respectively the first and the sixth best positions. These results contributed to spur interest in SF. More theoretical and experimental studies were soon published: Lederer and Guadarrama (2014) proposed an improved stopping criterion for SF when processing images relying on results from random matrix theory; Zhang et al. (2014) published an experimental comparison of six sparse coding algorithms, including SF; Romero et al. (2014) introduced a new algorithm inspired by SF with no meta-parameters; and Yang et al. (2014) modified the original SF algorithm by introducing a regularization penalty on the weight matrix. These studies highlighted a interest in the

[6] https://www.kaggle.com/c/challenges-in-representation-learning-the-black-box-learning-challenge

refinement and improvement of the original algorithm.

At the same time, on the more applied side, the simplicity of the SF algorithm favoured its adoption in many real-world applications. Different researchers deployed SF to tackle a variety of problems: Raja et al. (2015), Raja et al. (2016b), and Raja et al. (2016a) for iris recognition on smart-phones, for periocular image detection and for identifying biometric spoofing attacks; Chhabra and Dutta (2016) for iris detection; Rattani et al. (2016) for ocular verification; Raghavendra and Busch (2016) for periocular verification; Dong et al. (2014) and Dong et al. (2015) for vehicle type recognition; Hahn et al. (2015) for detecting human actions from videos; Yan et al. (2015) for recognizing driving posture; Lei et al. (2015) for intelligent fault diagnosis in mechanical apparati; Zhao and Feng (2017) for fault identification in planetary gearboxes; Gu et al. (2014) for assessment of image quality; Han and Lee (2014), Han and Lee (2016), Han et al. (2016) for detecting mistakes and over-blowing in flute playing and for instrument identification; Si et al. (2015) for estimating high-resolution images from low-resolution ones; Sun et al. (2016) for hyper-spectral anomaly detection; Lin et al. (2016) for electrical fault detection from infra-red images; Hach and Seybold (2016) for depth video denoising; Ortiz et al. (2016) for Parkinson detection; Zhang et al. (2016) for assessing the saliency of satellite images; Song et al. (2016) for hardware debanding; Liu et al. (2016b) for pedestrian detection; Liu et al. (2016a) for terrain classification using images from polarimetric synthetic aperture radars; Mei et al. (2017) for detecting Mura defect. All of these applications bear witness to the usefulness of SF and to the simplicity of integrating it and using it for very different machine learning tasks.

Some studies have also provided SF with some biological support. Bruce et al. (2016) analysed different biologically-grounded principles for representation learning of images, using SF as a starting point for the definition of new learning algorithms. Ryman et al. (2016) integrated SF in their system for modelling the olfactive temporal response in arrays of chemically diverse sensors. More interestingly, Kozlov and Gentner (2016) used SF to model the receptive fields of high-level auditory neurons in a songbird; relying on the idea that sparseness and normalization are canonical neural processing operations (Carandini and Heeger, 2012), their results show that SF can reproduce the receptive fields of the European starling and provide further support to the general hypothesis that sparsity and normalization are general principles of neural computation in the sensory system of animals. SF may then be of interest not only for practical machine learning applications, but also for modelling and understanding biological systems.

Having provided a complete description of FDL and SF, we can now move on to discuss the second specific topic of interest for this dissertation, that is, CSA.

2.4 Machine Learning under Covariate Shift

Sections 2.2.3 and 2.2.4 discussed standard machine learning problems and algorithms. These formulations relied on certain idealized modelling assumptions, most importantly the the i.i.d.

assumption. The present section reviews how the problem of learning changes when this assumption is dropped. Section 2.4.1 introduces the idea of *concept shift* to describe in general terms the violation of the i.i.d. assumption. Section 2.4.2 narrows the focus on a specific form of concept shift affecting the marginal distribution of the data, that is, *covariate shift*. Section 2.4.3 reviews different approaches suggested in the literature for tackling the problem of covariate shift, giving space to two approaches particularly relevant to this dissertation: CSA by *importance weighting* and CSA by *representation learning*, respectively.

2.4.1 Concept shift

In modelling machine learning problems many algorithms make the assumption of i.i.d. data. Unfortunately, this assumption may not always conform to reality. Indeed, sources of data may change over time and a single pdf may not describe accurately enough the data collected; for instance, the process generating the data may undergo a transformation due to a change in the external conditions, to an upgrade in the sampling resolution of a recording device, or to the introduction of new categorical labels.

Changes in one of the pdfs describing the data are generically defined as *concept shifts* (Webb et al., 2015). A taxonomy of different types of concept shifts may be provided along different dimensions: concept shift may be characterized by which pdf undergoes a shift, be it a marginal or a conditional pdf; or it may be described according to the dynamics of the shift, such as the magnitude of the shift, the transition dynamics (gradual, incremental, probabilistic), the shift duration (extended, abrupt, blip), or the periodicity (non-reoccurring, cyclical, non-cyclical) (Webb et al., 2015).

With respect to the affected pdf, the main types of concept shift are:

- *Covariate shift*, where the shift affects only the marginal distribution of the data, $p'(X) \neq p''(X)$, and where the notations $p'(X)$ and $p''(X)$ denote two pdfs at different times or over different subsets of data. Covariate shift also assumes that the conditional distribution of the labels given the data is not affected by the shift, $p'(Y|X) = p''(Y|X)$.
- *Virtual covariate shift*, where the shift affects the marginal distribution of the data, $p'(X) \neq p''(X)$, and, possibly, the conditional distribution of the labels given the data.
- *Prior probability shift*, where the shift affects only the marginal distribution of the labels, $p'(Y) \neq p''(Y)$.
- *Class drift*, where the shift affects the conditional distribution of the labels given the data, $p'(Y|X) \neq p''(Y|X)$.
- *Pure class drift*, where the shift affects the conditional distribution of the labels given the data, $p'(Y|X) \neq p''(Y|X)$, with the constraint that no shift happens on the marginal distribution of the data $p'(X) = p''(X)$.

Table 2.5 summarizes these forms of concept shift. Beside these main forms, it is possible to identify in the literature several other specific models of concept shift, such as *novel*

Covariate shift, Pure covariate shift	$p'(X) \neq p''(X)$ and $p'(Y\|X) = p''(Y\|X)$
Virtual concept shift	$p'(X) \neq p''(X)$
Prior probability shift	$p'(Y) \neq p''(Y)$
Class drift, Real concept drift	$p'(Y\|X) \neq p''(Y\|X)$
Pure class drift	$p'(Y\|X) \neq p''(Y\|X)$ and $p'(X) = p''(X)$

Table 2.5: Types of concept shift.

class appearance (Webb et al., 2015), *sample selection bias* (Heckman, 1977), *unbalanced data collection* (Japkowicz and Stephen, 2002), *source component shift* (Quiñonero-Candela, 2009), *probabilistic covariate shift* (Adel and Wong, 2015), *covariate shift with model misspecification* (Wen et al., 2014).

2.4.2 Covariate shift

Covariate shift[7] is one of the most important and studied forms of concept shift. Its relevance is due to the fact that changes in the marginal distribution of the data are quite common whenever the sampling conditions cannot be strictly controlled; while it is easy to follow a rigorous data collection protocol within the protected environment of a lab, it is extremely difficult to guarantee similar conditions in the wild. Covariate shift provides then a good model for several real-world settings, for instance scenarios where data are recorded from non-stationary sources (e.g.: biomedical signals such as EEG, Sugiyama and Kawanabe, 2012), where data are collected from different sets of users for training and test (e.g.: emotional speech processing, Schuller et al., 2010), or where data are acquired with different protocols (e.g.: image recognition, Torralba and Efros, 2011). In all these cases, training and test data can exhibit substantially different behaviours. For example, in the case of emotional speech processing (Schuller et al., 2010), training data collected from actors in a recording studio may have a significantly different distribution from test data collected from users on the street. This dissimilarity is due not only to environmental noise, but also to intrinsic discrepancies in the way sets of users express their own emotion. Thus, the distribution of the test data is not only affected by heterogeneous and potentially stronger noise, but it can also exhibit a different statistical behaviour or be defined on an entirely different domain. In such a situation, if we were to learn a discriminative model from the recordings in the studio, then the performances of the model on the recordings from the street would very likely be poor, as the statistical patterns captured by the model on the training data may not reflect the patterns in the test data. In order to learn a model from the training data that may generalize over the test data, some form of adaptation is required by

[7] The term *covariate* comes from the statistical lexicon; "covariate" is used to identify a variable that can be used for statistical prediction, in our case X. Alternatively, a covariate may be referred to as *independent variable* or as *explanatory variable*.

reshaping the distribution of the data or re-aligning their domains. When train and test data are adapted to fit a similar distribution, then a model able to process consistently all the data may be successfully trained using standard machine learning algorithms.

Learning under covariate shift means dropping the i.i.d. assumption, and setting up an inductive learning problem in which training data and test data are taken to have different distributions. Again, this is a very common case in reality, as the training environment of a learning machine will inevitably be different from the environment in which it will be deployed.

Formally, we adopt the following new assumption.

Assumption (Covariate Shift). *The marginal distribution of training and test data are different,* $p(X^{tr}) \neq p(X^{tst})$*, but the conditional distribution of the labels given the data is the same,* $p(Y|X^{tr}) = p(Y|X^{tst})$ *(Shimodaira, 2000).*

Notice that the final part of the assumption, which states the identity of conditional distributions, is crucial to making learning possible. If the training data and test data were to be sampled from two absolutely unrelated distributions, then no inference from the training data to the test data would theoretically be possible. The identity of the conditional distributions provides a "bridge" to transfer information about the training data to the test data (Sugiyama and Kawanabe, 2012). Thus, learning under covariate shift requires modelling the conditional distribution of the labels given the data $p(Y|X)$, taking into account the fact that the distributions of the training data $p(X^{tr})$ and test data $p(X^{tst})$ are different.

Notice that while the assumption of i.i.d. data is dropped, we normally retain an assumption of independent and identically distributed noise. Assuming that the noise is sampled independently from the same distribution both for the training and the test data allows us exclude the possibility that covariate shift may be due to a shift in the distribution of the noise; instead, the shift is expected to be due to more complex factors intrinsic to the process generating the training and the test data.

In covariate shift literature, the domain and the marginal pdf generating the training data are often referred to as the *source domain* $\mathbf{X^{src}}$ and *source pdf* $p(X^{src})$; the domain and the marginal pdf generating the test data are often referred to as the *target domain* $\mathbf{X^{tgt}}$ and *target pdf* $p(X^{tgt})$ (Margolis, 2011). For simplicity, we will keep the nomenclature of "training" and "test" domain and marginal, and we will use the term "target" with a slightly different meaning, as it will be explained later on. Also, the ability of a machine learning algorithm to learn despite covariate shift is sometimes referred to, in broad and informal terms, as *robustness*. However, we will refrain from using this term because of its generic meaning.

When learning under covariate shift, standard statistical guarantees on the performance of standard machine learning algorithms working on the i.i.d. assumptions are not valid any more (Vapnik, 1998; Bishop, 2007). New theoretical guarantees and bounds can be defined under the precise definition and qualification of covariate shift (Ben-David et al., 2010).

2.4.2.1 Measuring covariate shift

In order to work with covariate shift, it is necessary to define a measure to quantify it. So far, we have discussed covariate shift in purely qualitative terms, but it is only through the definition of a quantitative measure that we formalize its existence and may act on it.

As we are dealing with a difference between pdfs, an immediate solution for quantifying covariate shift is based on the use of a distance function between pdfs: $D : \mathfrak{P} \times \mathfrak{P} \to \mathbb{R}$ where $\mathfrak{P}$ is the space over which the pdfs are defined. Ideally, such a distance guarantees the properties of non-negativity, identity of the indiscernibles, symmetry and triangular inequality, all of which are required by a proper metric or distance function.

A standard measure from information theory (see Section 2.2.1) is the KL divergence, which is used to evaluate the distance between two pdfs defined on the same domain (MacKay, 2003):

$$D_{KL}\left[p\left(X'\right) \parallel q\left(X''\right)\right] = \int_{\mathcal{X}} p\left(X'\right) \frac{p\left(X'\right)}{q\left(X''\right)} dX.$$

Strictly speaking, $D_{KL}\left[\cdot \parallel \cdot\right]$ is a divergence and not a metric, as it does not satisfy the property of symmetry. It can be shown that the KL-divergence belongs to the family of f-divergences (Carter et al., 2007) and that, for the limit $p\left(X'\right) \to q\left(X''\right)$, the KL divergence is a good approximation of the Fisher metric for evaluating the distance between the pdfs on a statistical manifold (Amari, 2016). KL-divergence is thus a powerful descriptor for the distance between pdfs, but its estimation is challenging as it requires knowledge of the pdfs $p\left(X'\right)$ and $q\left(X''\right)$. It is possible to estimate $\hat{D}_{KL}\left[p\left(X'\right) \parallel q\left(X''\right)\right]$ through the estimation of the pdfs; in turn, $\hat{p}\left(X'\right)$ and $\hat{q}\left(X''\right)$ may be evaluated using a density estimation algorithm, such as *histogram-based density estimation*, *nearest neighbour density estimation*, or *kernel density estimation* (KDE) (Bishop, 2007). Density estimation however poses serious challenges, as the algorithms are very sensitive to the choice of parameters (such as number of bins in histogram-based density estimation, number of nearest neighbours in nearest neighbour density estimation, and kernel type and shape in kernel density estimation), and, in high dimensions, they require a large amount of samples to produce reliable results (Principe, 2010).

Alternative pdf measures based on entropies different from Shannon's may be defined by relying on the maximum entropy principle under specific conditions (Kapur, 1994) or relying on different divergences (Cha, 2007). However, a key problem remains that these distances often require a knowledge of the pdfs for their estimation.

A solution to the problem of estimating the pdfs is offered within the framework of information-theoretic learning proposed in Principe (2010). Exploiting the definition of *Renyi's quadratic entropy* $H_2\left[X\right] = -\log \int_{\mathcal{X}} p^2\left(X\right) dX$, it is possible to evaluate quadratic divergences, such as the Cauchy-Schwartz quadratic divergence, bypassing the explicit problem of estimating the pdfs, thanks to property of the Gaussian kernel:

$$\hat{D}_{CS}\left[p\left(X'\right) \parallel q\left(X''\right)\right] = 2\hat{H}_2\left[X'; X''\right] - \hat{H}_2\left[X'\right] - \hat{H}_2\left[X''\right],$$

where $\hat{H}_2[X';X'']$ is the *Renyi's quadratic cross-entropy* (Principe, 2010).

Statistical tests may also be used to estimate distances between pdfs. Strictly speaking, a statistical test provides only reasonable ground to reject or not reject a null hypothesis. As such, statistical tests may be used to assess whether two set of samples appear to come from the same pdf or not, but, rigorously, they do not normally provide a measure of distance. In other words, statistical tests may be used to evaluate the presence of covariate shift, but not to properly quantify it.

In the limit case of univariate pdfs, the (two-sample) Kolmogorov-Smirnov (KS) test provides a simple and reliable test to assess the equality of two pdfs (Duda et al., 2001). The KS test puts forward the null hypothesis that the samples from two distributions come from the same pdf. The KS statistic is computed as:

$$K(\mathbf{X}', \mathbf{X}'') = \sup_{\mathbf{X}', \mathbf{X}'' \in \mathcal{X}} \left| \hat{F}_p(\mathbf{X}') - \hat{F}_q(\mathbf{X}'') \right|,$$

where $\hat{F}_p(X')$ and $\hat{F}_q(X'')$ are the empirical cumulative distributions estimated from the first and the second set of samples. The null hypothesis is then rejected at level α if:

$$K > \sqrt{-\frac{1}{2} \log\left(\frac{\alpha}{2}\right) \frac{N_p + N_q}{N_p N_q}},$$

where N_p and N_q are the number of samples in the two sets.

The main limitation of the KS test is the fact that it works only with univariate data. Extension of the KS test to two dimensions have been suggested by Peacock (1983) and Fasano and Franceschini (1987) and carefully compared by Lopes et al. (2007). However, no direct simple generalization of the KS test to high dimensions is available.

Despite this, the KS test has sometimes been used to get a gross estimate of the similarity between multivariate pdfs $p(X'_1, X'_2, \ldots, X'_n)$ and $q(X''_1, X''_2, \ldots, X''_n)$ in high-dimensions by comparing through a KS test each individual dimension $p(X'_i)$ and $q(X''_i)$ (see, for instance, Hassan et al., 2013). This procedure is based on the intuition that, if the distribution on each dimension is assessed as identical by the KS test, $p(X'_i) = q(X''_i)$, then the overall multivariate distributions must be equal. However, this is not necessarily true, because a joint distribution $p(X'_1, X'_2, \ldots, X'_n)$ is uniquely defined not only by the marginal distributions $p(X'_i)$ but also by a *copula* function. For instance, the definition of two marginal univariate Gaussian pdfs is not sufficient to completely specify a joint bivariate Gaussian pdf, as infinite different bivariate Gaussians may be specified with different off-diagonal values in the covariance matrix. Assessing the similarity of each feature through a KS test is computationally very cheap, but not rigorously grounded. It may be used to get a very approximate evaluation of the multivariate pdfs, with the caveat that, if the marginal pdfs are different, then the joint pdfs are definitely different, but, if the marginal pdfs are the same, it is not possible to say the same about the joint pdf without knowledge or assumptions about the copula function.

A more rigorous solution is offered by the possibility of relying on the measure of Maximum

Mean Discrepancy (MMD) (Gretton et al., 2012). The MMD measure was introduced as a statistic for a distribution-free two-sample statistical test. Given two sets of samples $\mathbf{X}'$ and $\mathbf{X}''$ coming from two unknown distributions $p(X')$ and $q(X'')$, the aim of the statistical test is to evaluate the hypothesis that $p(X') = q(X'')$. The MMD test statistic is defined as:

$$MMD(\mathbf{X}', \mathbf{X}'') = \left\| \frac{1}{N_p} \sum_{i=1}^{N_p} \phi(\mathbf{x}'_i) - \frac{1}{N_q} \sum_{i=1}^{N_q} \phi(\mathbf{x}''_i) \right\|^2,$$

where $\phi(\cdot)$ is a function on a reproducing kernel Hilbert space (RKHS) that maximizes the difference between the expected value of $\phi(X')$ and the expected value $\phi(X'')$. Beside using this measure in a statistical test, Gretton et al. (2012) proved the double implication that $MMD(\mathbf{X}', \mathbf{X}'') = 0$ if and only if $p(X') = q(X'')$. The MMD statistic has then been adopted as a measure of distance between pdfs. In particular it has been used as a measure to be minimized or maximized in order to affect the learning of a pdf having a certain shape (see, for instance, Li et al., 2014).

Among the many pdf distances available, KL divergence, MMD distance and the distance approximation provided by the KS test will be the main distances we will rely on in this dissertation.

2.4.3 Covariate shift adaptation

The process of tackling covariate shift by aligning the distribution of the training data $p(X^{tr})$ and test data $p(X^{tst})$ take the name *covariate shift adaptation*. Alternatives nomenclature often used in place of CSA are *domain adaptation* and *transfer learning*. Domain adaptation underlines that an algorithm is designed to tackle a form of covariate shift mainly due to a difference in the domain of definition of the marginal distributions, $\mathcal{X}^{tr} \neq \mathcal{X}^{tst}$. Transfer learning underlines the fact that the aim of an algorithm is to transfer knowledge from a source domain to a target domain. Since the first nomenclature is more restrictive, and the second more generic (as it makes no reference to the type of concept shift that it tries to tackle), we will use the term CSA.

2.4.3.1 Measuring covariate shift adaptation

As in the case of covariate shift, in order to discuss CSA, it is necessary to define a way to quantify the degree of adaptation.

The simplest and most intuitive way to evaluate the degree of adaptation is to rely on the measures that we defined for covariate shift. By measuring the distance between pdfs before and after the use of CSA algorithms, it is possible to asses whether this distance decreased and to estimate the amount of this change. Specific measures for CSA have also been suggested, such as $\mathcal{H}$-divergence (Kifer et al., 2004; Ben-David et al., 2010). Distance measures are subject to the restrictions already described in the previous section for measuring covariate shift, that is, problem of computational complexity and reliability; especially when working in high

dimensions, it may be infeasible to get good estimates of the distance between the pdfs. It is convenient, then, to adopt proxy measures that could provide an approximate evaluation of the degree of CSA.

As CSA is often integrated in a supervised learning system, such as a classification system, an obvious way to assess the degree of CSA is by evaluating its contribution to the supervised task. If the underlying classifier system implicitly makes an assumption of i.i.d. data, the introduction of a CSA algorithm should sensibly improve the results. Indeed, without CSA, a classification system relying on an i.i.d. assumption would very likely under-perform due to the mismatch between the assumption about the data and the actual distribution of the data.

Let $\mathtt{A}$ be a generic classification system and $\mathtt{CSA} + \mathtt{A}$ the same classification system including a CSA module. It is possible to estimate the contribution of a CSA algorithm by evaluating two immediate quantities.

First, it is possible to evaluate how the performance changes classifying data coming from the test distribution $p\left(X^{tst}\right)$ using the base classifier $\mathtt{A}$ and the CSA-enriched classifier $\mathtt{CSA} + \mathtt{A}$. It is then possible to compute a performance percentage difference (PPD) as:

$$PPD = \frac{\mathcal{P}\left(\mathtt{CSA} + \mathtt{A}, \mathbf{X^{tst}}\right) - \mathcal{P}\left(\mathtt{A}, \mathbf{X^{tst}}\right)}{100 \cdot \mathcal{P}\left(\mathtt{A}, \mathbf{X^{tst}}\right)},$$

where the performance measure $\mathcal{P}\left(\cdot\right)$ is taken to be a function of the algorithm used and the data processed. High PPD values denote a good degree of CSA provided by the algorithm $\mathtt{CSA}$.

Alternatively, it is possible to evaluate how the performance changes by running $\mathtt{CSA} + \mathtt{A}$ on data coming from the training distribution $p\left(X^{tr}\right)$ and on data coming from the test distribution $p\left(X^{tst}\right)$. The effect of CSA may be estimated computing the percentage drop (PD) suggested by Torralba and Efros (2011):

$$PD = \frac{\left(\mathcal{P}\left(\mathtt{CSA} + \mathtt{A}, \mathbf{X^{tr}}\right) - \mathcal{P}\left(\mathtt{CSA} + \mathtt{A}, \mathbf{X^{tst}}\right)\right) \cdot \mathcal{P}\left(\mathtt{CSA} + \mathtt{A}, \mathbf{X^{tst}}\right)}{100},$$

or the cross-domain difference (CD) proposed by Tommasi et al. (2015):

$$CD = \frac{1}{1 + \exp^{\left(\mathcal{P}\left(\mathtt{CSA}+\mathtt{A}, \mathbf{X^{tr}}\right) - \mathcal{P}\left(\mathtt{CSA}+\mathtt{A}, \mathbf{X^{tst}}\right)\right)/100}}.$$

Low PD or CD values mean that, after performing CSA on data coming from different distributions, the classification algorithm can achieve a similar performance on all the data.

It is important to point out that all these measures (PPD, PD and CD) are measures that are relative to a base performance, either the performance without CSA (for PPD) or the performance on data coming from the training distribution (for PD and CD). This means that the range of values of these performance indexes are bounded by the base performance. Care must be taken when comparing these measures across different algorithms, as the base performance may change for different algorithms.

Notice also that, for these relative proxy measures to be accurate and meaningful, it would be necessary to show that the change in performance is due only to CSA and not to other

transformations performed inside the CSA algorithm; this, however, is a condition very difficult to validate, as a CSA algorithm may perform also other types of transformations, such as compression, projection or sparsification. Despite this conceptual limitation, these measures offer an approximate but practical estimation of the degree of CSA, and in the following chapters we will make use of relative measures, especially PPD.

2.4.3.2 Taxonomy of covariate shift adaptation algorithms

In order to perform adaptation between training and test data, a CSA algorithm needs to be able to access data from the distribution it is aiming to adapt to. We thus expect to have data from the test pdf at adaptation time. It is possible to partition the data for adaptation as follows:

- *Training data:* data from the training distribution used for adaptation, $\mathbf{X^{tr}} \sim p(X^{tr})$;
- *Target data:* data from the test distribution used for adaptation, $\mathbf{X^{tgt}} \sim p(X^{tst})$.

Target data constitute a sub-set of samples taken from the same pdf as the test data, which will only be used for adaptation and not for evaluation.

If available, an adaptation algorithm may rely on the labels that are provided along with the training data. It is possible then to distinguish two main forms of adaptation. The first one is *unsupervised adaptation*, which uses only unlabelled data $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$; this form of adaptation can rely only on the estimation of the marginal distributions $p(X^{tr})$ and $p(X^{tst})$ and it can try to compensate for their difference. The second form is *supervised adaptation*, which exploits both data and available labels $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$; this form of adaptation can rely not only on the estimation of the marginal distributions $p(X^{tr})$ and $p(X^{tst})$, but also on the estimation of the conditional distribution of the labels $p(Y|X)$.

Now, even if CSA is represented as a stand-alone process, it is important to repeat that an adaptation algorithm is usually instrumental for more general-purpose learning tasks. Indeed, CSA algorithms are often designed to be integrated in machine learning systems tackling one of the specific forms of learning problems discussed in Section 2.2.4. A CSA algorithm may be directly embedded in an unsupervised learning system in order to lead to the learning of new representations that are not affected by covariate shift. It may also easily fit in a learning system based on the semi-supervised learning paradigm; labelled and unlabelled samples may be jointly exploited for carrying out CSA, while labelled data may be used to learn a supervised function. More commonly, however, CSA is integrated in a supervised learning system; in this case, the explicit aim of CSA is to compensate for covariate shift in order to ease an ensuing supervised learning task.

Being the most widespread application of CSA, we will focus our attention on CSA systems designed for classification. It is worth underlining that supervised discriminative algorithms, such as classification systems, are affected by covariate shift, even if they apparently model only a conditional distribution $p(Y|X)$. According to our assumptions, the conditional distribution is the same for both the training and the test data and, as such, the conditional distribution learned from the training data should be automatically generalizable to the test data. However,

	Unsupervised Induction	***Semi-Supervised Induction***	***Supervised Induction***
Unsupervised Adaptation	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}\}$ Model Selection: $\{\mathbf{X^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}\}$	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$ Model Selection: $\{\mathbf{X^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}, \mathbf{Y^{tst}}\}$	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ Model Selection: $\{\mathbf{X^{tgt}}, \mathbf{Y^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}, \mathbf{Y^{tst}}\}$
Supervised Adaptation	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}\}$ Model Selection: $\{\mathbf{X^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}\}$	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$ Model Selection: $\{\mathbf{X^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}, \mathbf{Y^{tst}}\}$	Adaptation: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}, \mathbf{X^{tgt}}\}$ Induction: $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ Model Selection: $\{\mathbf{X^{tgt}}, \mathbf{Y^{tgt}}\}$ Evaluation: $\{\mathbf{X^{tst}}, \mathbf{Y^{tst}}\}$

Table 2.6: Types of CSA.

this is not the case, because many supervised algorithms optimize an average error estimated on the training domain (Yamada et al., 2012). Standard supervised algorithms, such as SVM, are affected by the density of the training data; only purely conditional models, such as Gaussian processes, are immune to covariate shift (Quiñonero-Candela, 2009). It is therefore important, even when learning conditional distribution through standard supervised algorithms, to perform CSA.

When learning for classification, we will again expect to have data from the training and the test distribution. It is possible then to partition the data for classification as follows:

- *Training data:* data from the training distribution used for learning, $\mathbf{X^{tr}} \sim p\left(X^{tr}\right)$;
- *Target data:* data from the test distribution used for model selection, $\mathbf{X^{tgt}} \sim p\left(X^{tst}\right)$;
- *Test data:* data from the test distribution used for evaluation, $\mathbf{X^{tst}} \sim p\left(X^{tst}\right)$.

Notice that the training data and the target data are the same as in adaptation, but they are used now in a different way. The training data with its associated labels $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ will be used for learning a classification model. The target data with its associated labels $\{\mathbf{X^{tgt}}, \mathbf{Y^{tgt}}\}$ will be used for model selection; thus target data take the place of what previously was called validation data. The reason to rely on data coming from the test distribution to perform model selection is that we do not want to select a model that is tied to the specific training distribution $p\left(X^{tr}\right)$ but one that can work well for the test distribution $p\left(X^{tst}\right)$ too. Finally, the test data with its associated labels $\{\mathbf{X^{tst}}, \mathbf{Y^{tst}}\}$ will be used for performance evaluation; having a subset of data from the test distribution used for evaluation only allows us to properly evaluate the generalization performance of the overall machine learning system.

Table 2.6 illustrates the combination of different types of adaptation (supervised, unsupervised) with different types of inductive learning problems (unsupervised, semi-supervised, supervised). As explained, our attention is restricted to the last column, that is, to unsupervised and supervised CSA algorithms applied to classification.

2.4.3.3 Covariate shift adaptation algorithms in the literature

Several different techniques for CSA have been proposed in the literature. A high-level categorization of CSA algorithms may be based on the following families of algorithms (Jiang, 2008; Marsella et al., 2010):

- *Representation learning algorithms (RL):* RL algorithms can be used to perform CSA by projecting data samples onto new learned representations in a space where the distance between the marginal distributions of the training and test data is minimized.
- *Importance weighting algorithms (IW):* IW algorithms are based on the idea of rescaling the learning process by taking into account the ratio between the pdf of the training data $p(X^{tr})$ and the pdf of the test data $p(X^{tst})$.
- *Ensemble methods:* Ensemble methods include different techniques (such as *bagging* or *boosting*) designed to combine together several classifiers. Different classifiers can be trained on various subsets of training and test data and their outputs may be combined and scaled in order to compensate for the difference in the original marginal distributions.
- *Bayesian priors methods:* Bayesian priors methods rely on Bayesian learning algorithms, and they explicitly try to compensate for covariate shift by introducing prior knowledge in the prior probability of a model (Li and Bilmes, 2007).
- *Self-labelling methods:* Self-labelling methods are grounded in the idea of compensating for covariate shift through an iterative approach based on learning a model using labelled training data, labelling the unlabelled target data, and then re-training the model using a subset of data sampled from both the labelled training data and the newly-wed target data. Repeating this procedure allows one to refine the learned model and progressively align the marginal distributions of the training and the test data (Blum and Mitchell, 1998).
- *Cluster-based methods:* Cluster-based methods are based on the idea of learning manifolds of data across high-density regions of space that span both the training and the target domain. Clustering allows for the discovery of regions of high-density that can explain both the training and the test data (Margolis, 2011).

These categories define a coarse taxonomy that can be used to classify several algorithms for CSA. However, given the vague boundaries between CSA and other forms of learning, such as transfer learning or multi-task learning, it is hard to come up with an exhaustive taxonomy able to account for all the algorithms presented in the literature. In the following, we will focus on two families that have gathered a significant amount attention and which have been successfully applied to a wide range of problems: CSA via RL and CSA via IW.

2.4.3.4 Covariate shift adaptation via representation learning

As discussed in Section 2.2.4, representation learning is a generic form of unsupervised learning aimed at discovering a mapping $f : \mathcal{X} \rightarrow \mathcal{Z}$ projecting the original representations onto new useful and high-level learned representations.

RL for CSA aims at exploiting existing algorithms in order to integrate into their objective a compensation for the covariate shift between training data and test data. Thus, RL for CSA tries to merge two aims: (i) learn new and better representations $\mathbf{Z}$; (ii) tackle the shift in the pdfs of the training data and test data. The second objective may be formally expressed in the requirement of discovering a new space $\mathbb{R}^L$ such that:

$$D\left[p\left(X^{tr}\right), p\left(X^{tst}\right)\right] > D\left[p\left(Z^{tr}\right), p\left(Z^{tst}\right)\right].$$

RL algorithms may range from simple feature selection algorithms looking for covariate shift-minimizing sub-spaces (such as feature sub-setting using conditional probability models, Satpal and Sarawagi, 2007) to more refined algorithms discovering new latent spaces that reduce the effect of covariate shift.

Most of the existing RL algorithms work with the assumptions of i.i.d. data and need to be redesigned to work under the assumption of covariate shift. However, in some cases, such as with DAE (Vincent et al., 2008b), it may be argued that simple RL algorithms are able to learn robust representations that can perform CSA (Bengio et al., 2012). A DAE module computes a new representation $\mathbf{z}_i$ of a data sample $\mathbf{x}_i$ as:

$$\mathbf{z}_i = f\left(\mathbf{W}\tilde{\mathbf{x}}_i + \mathbf{b}\right),$$

where $\tilde{\mathbf{x}}_i$ is a corrupted noisy version of $\mathbf{x}_i$, $f : \mathbb{R} \to \mathbb{R}$ is an element-wise non-linear function, $\mathbf{W} \in \mathbb{R}^{L \times M}$ is the weight matrix and $\mathbf{b} \in \mathbb{R}^L$ is the bias vector. From the learned representation $\mathbf{z}_i$ the DAE computes a reconstruction as:

$$\hat{\mathbf{x}}_i = g\left(\mathbf{V}\mathbf{z}_i + \mathbf{c}\right),$$

where $g : \mathbb{R} \to \mathbb{R}$ is an element-wise non-linear function, $\mathbf{V} \in \mathbb{R}^{M \times L}$ is the reconstruction weight matrix and $\mathbf{c} \in \mathbb{R}^M$ is the reconstruction bias vector. According to the implementation of the DAE, the non-linear functions might be the same, $f(\cdot) = g(\cdot)$, or the weight matrices might be tied, $\mathbf{W} = \mathbf{V}^\top$. Learning is performed by minimizing over all the parameters a reconstruction loss, such as the square error:

$$\underset{\mathbf{W}\in\mathbb{R}^{L\times M},\mathbf{V}\in\mathbb{R}^{M\times\mathbb{R}^L},\mathbf{b}\in\mathbb{R}^L,\mathbf{c}\in\mathbb{R}^M}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\hat{\mathbf{x}}_i - \mathbf{x}_i\right)^2.$$

A solution to this optimization problem can be computed by gradient descent minimizing over all the available data. Even if no rigorous theoretical explanation has been provided to justify the use of DAE for CSA, DAE and stacked DAE (Vincent et al., 2010) were demonstrated to be able to perform effective CSA (Glorot et al., 2011).

A relevant RL algorithm explicitly designed for CSA is *sub-space alignment* (SSA) (Fernando et al., 2013). SSA works under the assumption that the space defined by the PCA components of the training data can be projected onto the space of the PCA components of the test data. SSA first computes the PCA spaces of the training data and the test data, and then it tries to

align them computing the following representations:

$$\mathbf{Z^{tr}} = \mathbf{X^{tr}}\mathbf{T}\mathbf{T}^{\top}\mathbf{U}$$
$$\mathbf{Z^{tst}} = \mathbf{X^{tst}}\mathbf{U},$$

where $\mathbf{T}$ is the matrix of the eigenvectors of the covariance matrix for the training data, and $\mathbf{U}$ is the matrix of the eigenvectors of the covariance matrix for the test data.

Other approaches to RL algorithms include methods based on the identification of pivot features (such as *structural correspondence learning*, Blitzer et al., 2006), methods grounded in manifold learning (such as *geodesic flow kernels*, Gong et al., 2012), and methods based on the minimization of the differentiable measure of distance between the original and the learned distribution (such as *kernel mean matching* (KMM), Huang et al., 2007; Quadrianto et al., 2009).

In general, CSA through RL may be a good choice when the pdf $p(X)$ of the original data is particularly complex or noisy with respect to the labels $\mathbf{Y}$; if the conditional distribution $p(Y|X)$ is ill-behaved, learning a new intermediate representation Z may ease not only the task of CSA but the task of classification as well. CSA through representation learning may then provide new representations that solve the problem of covariate shift and produce a better-behaved conditional distribution $p(Y|Z)$ at the same time. The RL approach has been successfully applied to many problems dealing with complex data, such as image recognition (Kulis et al., 2011; Tzeng et al., 2014), sentiment analysis (Glorot et al., 2011; Li et al., 2014) and emotional speech recognition (Deng et al., 2014).

2.4.3.5 Covariate shift adaptation via importance weighting

IW is a widely-adopted technique for performing CSA based on the idea of rescaling the loss function of a learning algorithm by the ratio between the distribution of the training data and the test data. This ratio is meant to emphasize the contribution to learning of training points close to the test points and to discount the contribution of training points falling far from the test points. IW works under the assumption that the training pdf and the test pdf are defined on the same domain (thus guaranteeing the possibility of computing the ratio) and on the assumption that the conditional distribution exhibits a continuous behaviour over the training domain and the test domain.

Given a learning algorithm with a loss function $\mathcal{L}\left(\mathbf{X^{tr}}\right)$ optimized over the training data $\mathbf{X^{tr}}$ with respect to the set of parameters Θ, IW redefines the loss function as:

$$\mathcal{L}\left(\mathbf{X^{tr}}, \mathbf{X^{tst}}\right) = \frac{p\left(X^{tst}\right)}{p\left(X^{tr}\right)}\mathcal{L}\left(\mathbf{X^{tr}}\right),$$

where $\frac{p\left(X^{tr}\right)}{p\left(X^{tst}\right)}$ is the importance weight which evaluates the ratio between the pdf of the test data and the training data. More refined forms of importance weighting are adaptive importance weighting (adding a flattening parameter γ to the importance) and regularized weighting (introducing a regularization term). Many traditional machine learning algorithms may be converted

to their importance-weighting counterparts, such as importance-weighting least squares (IWLS), importance-weighting logistic regression (IWLR) or importance-weighting SVM (IWSVM) (Sugiyama and Kawanabe, 2012).

A crucial step in applying IW is the estimation of the ratio $\frac{p(X^{tr})}{p(X^{tst})}$. A first naive solution is based on the evaluation of this ratio through the estimation of the individual pdfs through KDE. However this approach is very fragile due to the limitations of KDE (see Section 2.4.2). Other methods allow for a direct evaluation of the ratio of the pdfs $\frac{p(X^{tr})}{p(X^{tst})}$, side-stepping the intermediate problem of estimating the individual pdfs. These approaches include the use of KMM, measures of distance between the pdfs such as KL-divergence (Kullback-Leibler importance estimation procedure, KLIEP) or least squares (least-squares importance fitting LSIF), or the re-casting of the ratio estimation problem as a decision problem (Sugiyama and Kawanabe, 2012).

The IW approach has been studied in the statistics literature and it is theoretically well-grounded. It has been widely used for CSA, but also for tackling other forms of concept shift, like class imbalance. However, IW also has some critical shortcomings, including the computational complexity for estimating the pdfs or their ratio, and the fact that the direct processing of the original representations does not include any explicit representation learning. Indeed, differently from CSA via RL, CSA via IW does not learn new representations. As such, IW-based algorithms tackle exclusively the problem of covariate shift, without learning a possibly better-behaved conditional distribution $p(Y|Z)$. Despite this, the IW approach has been successfully applied to many problems, such as speaker identification, EEG processing, natural language processing (Sugiyama and Kawanabe, 2012) and emotional speech recognition (Hassan et al., 2013).

Having concluded the presentation of the CSA problem in machine learning, in the next section we discuss the open problems that will be addressed in the remaining chapters of this dissertation.

2.5 Challenges and Problems Addressed in This Work

In this chapter, we started by providing a discussion of the problem of learning and then we moved on to provide a more formal and complete description of the field of machine learning. Within this field, we have focused on two areas of interest: FDL and CSA. In this section, we review why these two areas are particularly significant and what theoretical and practical challenges they present.

In Section 2.3, we presented FDL as a newly-developed and promising area of research in unsupervised learning. FDL introduced an innovative and compelling approach to representation learning, and the results achieved by the SF algorithm led to a wide-spread adoption of this algorithm. However, despite this success, some crucial aspects of FDL and SF remain unexamined. The very definition of FDL algorithms is intuitively clear, but not very rigorous. This lack of formality makes it hard to draw the boundaries of the class of FDL algorithms

and to understand which underlying features are actually common to all the FDL algorithms. This open problem justifies our first research question: *how can we rigorously define a FDL algorithm?* (see Section 1.2). Once we move from the study of the abstract class of FDL algorithms to the concrete instance of the SF algorithm, we are bound to face unavoidable questions about the inner working of this algorithm. The wide-spread use of SF in many applied scenarios is due mainly to its practical success more than to any deep understanding or theoretical justification. This may actually give us a biased view of the usefulness of SF (as we are aware of the cases in which it successfully worked, but we do not know anything of all the instances in which it did not produce good results) and it does not help us to understand why and under which conditions the algorithm is effective. This leads us to a scientific inquiry informed by the following research question: *can we explain the behaviour of the SF algorithm through the lens of our redefined conceptual and theoretical understanding of FDL algorithms?* (see Section 1.2). Defining FDL algorithms and explaining the SF algorithm is just the first step in studying and developing new FDL algorithms. The simplicity and the efficiency of SF naturally invites us to consider the possibility of developing new FDL algorithms that may be applied to alternative scenarios. This opens up a vast potential area of research. Without focusing on the practicalities of defining concrete new algorithms, we consider instead the challenge of studying the requirements for doing so and sketching the guidelines that may inform the process of creating new algorithms. This challenge is expressed in our research question: *building upon our newly-developed understanding of SF algorithms, can we identify, explain or design alternative FDL algorithms?* (see Section 1.2). These open problems will be the topic of Chapter 3.

In Section 2.4, we discussed the particularly sensitive problem of covariate shift. This problem constitutes a critical challenge for the development of machine learning algorithms that can learn from a given set of data and that can be then deployed in complex and subtly-different environments. CSA has drawn significant attention, and, even if several potential solutions have been proposed, this problem is far from being solved. Solutions are often specific to given problems and they may have high computational requirements. Finding alternative simple and general solutions is currently an open problem. From the point of view of simplicity, FDL algorithms seem particularly suited for dealing with covariate shift. As minimal theoretical study has been done on FDL algorithms, there is space for investigating whether these algorithms could be successfully used for CSA. This unexplored open problem is defined by our research question: *can FDL be successfully applied to CSA, and, if so, how?* (see Section 1.2). Answering this question on a theoretical level leads us to consider which concrete implementations of FDL algorithms could be used to perform CSA. Not surprisingly, we choose as a starting point the SF algorithm. This choice allows us to rely on the understanding of the algorithm that we have developed, and to extend it to the covariate shift scenario. The challenge we face is to define the conditions and the limits within which SF can perform successful CSA. This problem is condensed in our research question: *can we perform CSA via SF?* (see Section 1.2). Studying the SF algorithm we are led to uncover its limitations in performing CSA. The shortcomings

of SF automatically produce a new challenge: whether and how such limitations may be addressed. This brings us to consider more concrete open problems related with covariate shift and to try to address them using FDL algorithms. Such a challenge is what motivates our last main research question: *can we find a FDL algorithm that can overcome the limitations of SF?* (see Section 1.2). These open problems will be the topic of Chapter 4.

Chapter 3

Feature Distribution Learning

This chapter provides an in-depth study of the recent and promising family of FDL algorithms. The aim is to offer a clear understanding of what FDL algorithms are and to suggest which already existing algorithms qualify for this category. Moreover, an additional objective of this chapter is to shed light on the dynamics of FDL algorithms, in particular to provide an explanation for the successful but so-far-unexplained SF algorithm.

Section 3.1 begins by providing a new and more precise definition of FDL algorithms using optimization and information-theoretic terms. Section 3.2 provides a close review of the most representative algorithm for FDL, that is SF. Section 3.3 continues with a rigorous theoretical analysis of SF aimed at proving the properties and the bounds of SF. Section 3.4 validates the theoretical conclusions on SF using synthetic and real-world data sets. Section 3.5 considers alternative FDL algorithms and discusses the extension of our conclusions to these new models. Finally, Section 3.6 concludes the chapter by summing up all the results presented.

3.1 Conceptual Analysis of Feature Distribution Learning

This section provides a conceptual analysis of distribution learning. We start with a *pars destruens* in which we demonstrate the limitations of the standard definition of distribution learning and we conclude with a *pars construens* in which we propose a new alternative definition of FDL.

Section 3.1.1 highlights the limitation and the problems connected with the use of the standard intuitive definition of FDL. Section 3.1.2 suggests the use of information-theoretic and optimization concepts to define distribution learning. Finally, Section 3.1.3 proposes an alternative definition of DDL and FDL which will be used in the following sections and will inform our theoretical analysis.

3.1.1 Limits of the standard definition of distribution learning

The preliminary definition of FDL offered first in Section 2.2.4 and in Section 2.3 is, at a deeper analysis, unsatisfactory. Both the simple definition of DDL as the learning of the true distribution that underlies the generation of the data and the definition of FDL as the learning of a

distribution with a set of desirable properties present critical issues. On one side, these definitions cannot really be used to categorize algorithms that both try to learn the true distribution of the data $p(X)$ and model a useful distribution of the features $p(Z)$, such as sparse RBM (Ranzato et al., 2007). On the other side, it is not clear what is implied by the statement that FDL ignores learning the true data distribution $p(X)$; "ignoring" may mean anything ranging from the extreme option of "completely disregarding the problem of learning the data distribution" to the more moderate option "not caring about optimizing the learning of the data distribution".

3.1.2 Infomax principle and informativeness principle

A better understanding of FDL is required to properly study and analyse concrete instances of FDL, such as SF. Vincent et al. (2010) argued that an unsupervised learning algorithm can generate good representations by satisfying two requirements: (i) retaining information about the input, and (ii) applying constraints that lead to the extraction of useful information from noise.

In more general terms, we conjecture that good unsupervised representations may be obtained by satisfying the two following information-theoretic requirements: (i) maximizing the mutual information between input and output, and (ii) maximizing a measure of information of the output. The first requirement is the same as the one stated by Vincent et al. (2010), and it corresponds to the *infomax principle* (Linsker, 1989). The second requirement, for lack of a better term, will be referred to as *informativeness principle*. Notice that this balance between different information-theoretic properties makes perfect sense with respect to the limits of optimizing single individual information-theoretic quantities, as we discussed in Section 2.2.1.

According to this understanding, the aim of FDL may be expressed as a pure optimization of the informativeness principle. That is, it tries to learn a map $f : \mathcal{X} \to \mathcal{Z}$ such that the pdf $p(Z)$ of the new representations $\mathbf{Z}$ has maximal informativeness. Maximizing the informativeness may be simply expressed as the minimization of the entropy $H_S[Z]$ or the maximization of the relative entropy between the learned pdf $p(Z)$ and the entropy-maximizing pdf $q(Z)$, that is, $D_{KL}[p(Z) \parallel q(Z)]$.

However, without other requirements, the objective of maximizing the information is not sufficient to lead to any useful or meaningful learning. The optimal solution of the problem of maximizing $H_S[Z]$ is trivially learning a pdf with the shape of a Dirac delta function. If all the original samples $\mathbf{x}_i$ are mapped onto an arbitrary representation $\bar{\mathbf{z}}_i$, then the pdf $p(Z)$ will have the shape of a Dirac delta function centred on $\bar{\mathbf{z}}_i$. This pdf has minimal entropy and, therefore, maximal informativeness. However, it is clear that arbitrarily mapping all the samples $\mathbf{x}_i$ to a constant representation $\bar{\mathbf{z}}_i$ has no point: it would maximize the information in $p(Z)$ but discard all the information carried by $p(X)$.

It is necessary, therefore, to take into consideration the infomax principle, as well. This translates into the requirement of the maximization of the mutual information $MI[X;Z]$ or,

equivalently, to the maximization of the relative entropy between $p(X,Z)$ and $p(X)\,p(Z)$: $D_{KL}\left[p(X,Z) \parallel p(X)\,p(Z)\right]$.

We then conjecture that, like any unsupervised learning algorithm, FDL must somehow negotiate the trade-off between the infomax principle and the informativeness principle:

$$\max_{p(Z)\in\mathfrak{P}} \underbrace{D_{KL}\left[p(X,Z) \parallel p(X)\,p(Z)\right]}_{\substack{\text{infomax}\\ \text{principle}}} + \underbrace{D_{KL}\left[p(Z) \parallel q(Z)\right]}_{\substack{\text{informativeness}\\ \text{principle}}}. \tag{3.1}$$

Thus, even if the definition of FDL makes no reference to the infomax principle, it must be taken into account in some way.

The learning objective defined in Equation 3.1 is bound to remain mainly theoretical. Information-theoretic quantities, like relative entropy, are hard to evaluate and therefore it is necessary to rely on approximations or heuristics. Moreover, the optimization of an objective function composed by multiple terms is often challenging. Operational research suggests that a simpler approach to optimize two objective terms would be to translate one objective into a constraint and explicitly optimize the remaining term.

3.1.3 Alternative definitions of distribution learning

Relying on the generic definition of an unsupervised learning algorithm as an algorithm solving the optimization problem in Equation 3.1, we can now put forward new alternative definitions for DDL and FDL.

Data distribution learning. In relation to Equation 3.1, we can define DDL as any unsupervised learning algorithm whose main objective is maximizing the infomax principle (first term in Equation 3.1), while the maximization of the informativeness principle (second term in Equation 3.1) is accounted through priors or constraints.

For instance, DAEs approximate the maximization of the mutual information $MI[X;Z]$ through the minimization of the reconstruction error, as Vincent et al. (2010) proved by showing that minimizing the reconstruction error is indeed equivalent to maximizing a lower bound on the mutual information. Similarly, RBMs approximate the maximization of the mutual information $MI[X;Z]$ through the minimization of the divergence between the distribution of the data and the learned distribution via the maximization of the log probability of the data (Hinton et al., 2006). On the other hand, these algorithms do not tackle the problem of maximizing the relative entropy $D_{KL}\left[p(Z) \parallel q(Z)\right]$ directly, but they often address it using constraints (such as implementing bottleneck architectures, Tishby et al., 2000) or imposing priors (such as adding a sparsity penalty to the learning objective, Vincent et al., 2010).

Feature distribution learning. Again, in relation to Equation 3.1, we can define FDL as any unsupervised learning algorithm whose main objective is maximizing the informativeness principle (second term in Equation 3.1), while the maximization of the infomax principle (first

term in Equation 3.1) is accounted through priors or constraints.

Relying on this new understanding of distribution learning, we can move on to describe concrete FDL algorithms, and in the next section we will provide a first in-depth review of SF.

3.2 Analysis of the Sparse Filtering Algorithm

In this section we review the most emblematic algorithm for FDL, that is SF. We analyse the SF algorithm in order to provide a first basic understanding of the algorithm, upon which we will build a more rigorous theoretical analysis.

Section 3.2.1 discusses what forms of sparsity the SF algorithm tries to learn. Section 3.2.2 analyses the SF algorithm, decomposing it into its constituent steps. Section 3.2.3 comments on SF, highlighting a couple of important properties of immediate derivation.

3.2.1 Enforcement of sparsity in sparse filtering

The primary aim of SF is to learn a pdf $p(Z)$ which maximizes the sparsity of the learned representations $\mathbf{z}_i$. SF achieves this objective by enforcing three properties (Ngiam et al., 2011) on the matrix of learned representations $\mathbf{Z}$ (see Section 2.3.3):

- *Population sparsity*: each sample $\mathbf{z}_i$ is required to be sparse, that is, have a low activation computed as: $\ell_1(\mathbf{z}_i) = \sum_{j=1}^{L} |z_{i,j}|$.
- *Lifetime sparsity*: each feature $\mathbf{z}_{\cdot,j}$ is required to be sparse, that is, have a low activation computed as: $\ell_1(\mathbf{z}_{\cdot,j}) = \sum_{i=1}^{N} |z_{i,j}|$.
- *High dispersal*: all the features $\mathbf{z}_{\cdot,j}$ are required to have approximately the same activation, computed as the variance $Var[\ell_1(\mathbf{z}_{\cdot,j})]$. Lower variances correspond to higher dispersal.

As shown by Ngiam et al. (2011), the enforcement of these three properties translates into learning non-degenerate sparse representation (see Section 2.3.3).

3.2.2 Implementation of sparse filtering

Practically, SF is implemented as a simple algorithm in six steps: (refer to Figure 3.1 for an illustration of this decomposition and to Figure 3.2 for an illustration of the transformations on the data):

A0. Initialization of the weights: the weight matrix $\mathbf{W}$ with dimensions $\mathbb{R}^{L \times M}$ is initialized by sampling each component from a normal distribution $\mathcal{N}(0,1)$.

A1. Linear projection of the original data: $f_{A1}(\mathbf{X}) = \mathbf{W}\mathbf{X}$. The weight matrix $\mathbf{W}$ can be interpreted as a *dictionary* (Denil and de Freitas, 2012) or a *filter bank* (Dong et al., 2015), where each row is a codeword or a filter applied to each sample in the columns of $\mathbf{X}$. While training, SF tries to learn a simple linear transformation of $\mathbf{X}$, which, in non-degenerate cases, amounts to a scaling and/or rotation of the original data. Notice that

a linear transformation is a conservative transformation which, given proper conditions (Dasgupta and Gupta, 2003), preserves the original neighbourhood structure of the data $\mathbf{X}$. Refer to Figure 3.2(a) and 3.2(b) for an illustration of this transformation.

A2. Non-linear transformation: $\mathbf{F} = f_{A2}(\mathbf{WX})$, where $f_{A2} : \mathbb{R} \to \mathbb{R}$ is an element-wise non-linear function. Although this non-linear function can, in principle, be arbitrarily chosen, all the implementations available in the literature used an element-wise absolute-value function $f_{A2}(x) = |x|$. For practical reasons, this non-linearity is implemented as a soft absolute-value function $f_{A2}(x) = \sqrt{x^2 + \epsilon}$ where ϵ is a small negligible value (for instance, $\epsilon = 10^{-8}$). The practical reason to adopt this soft-thresholding version is that it prevents any value from being identically zero, which could cause potential division-by-zero errors in the following steps of the algorithm. Notice that a soft absolute-value function is defined as $f_{A2} : \mathbb{R} \to \mathbb{R}_{>0}$. This means that every sample $\mathbf{z}_i$ in the Cartesian space $\mathbb{R}^L$ after transformation A1 is re-mapped into the positive orthant of the the same space $\mathbb{R}^L$. This transformation can be interpreted as a rigid folding of the space, thanks to which successive transformations will be applied equally to data in all the different orthants (Montufar et al., 2014). The projection in the positive orthant is crucial to perform normalizations in the following steps of SF. Refer to Figure 3.2(b) and 3.2(c) for an illustration of this transformation.

A3. ℓ_2-normalization along the features (or along the rows): $\tilde{\mathbf{F}} = f_{A3}(\mathbf{F}) = \frac{f_{i,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}}$. In this step, each feature $\tilde{\mathbf{f}}_{\cdot,j}$ is normalized so that its squared activation is one, $\sum_{i=1}^{N} f_{i,j}^2 = 1$. This property corresponds to the requirement of high dispersal (Ngiam et al., 2011; Springenberg and Riedmiller, 2012); indeed constraining the second moment of each feature amounts to constraining and limiting their variance. Geometrically, the ℓ_2-normalization along the rows can be interpreted in two ways. If the samples are plotted in the feature space, then this step corresponds to a rescaling of the axes. See, for illustration, Figure 3.2(c) and 3.2(d) where the passage from the third to the fourth plot is just a rescaling of the axes. This perspective can hint to the fact that this transformation does not substantially alter the structure of the data. If the features are plotted into the sample space instead, this step corresponds to a projection of the features on a unit hypersphere.

A4. ℓ_2-normalization along the samples (or along the columns): $\mathbf{Z} = \hat{\mathbf{F}} = f_{A4}\left(\tilde{\mathbf{F}}\right) = \frac{\tilde{f}_{i,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}}$. In this step, each sample $\hat{\mathbf{f}}_i$ is normalized so that its squared activation is one, $\sum_{j=1}^{L} \hat{f}_{i,j}^2 = 1$. Geometrically, the ℓ_2-normalization along the columns can be interpreted in two ways. If the samples are plotted in the feature space, this step can be interpreted as a projection of the samples on a unit hypersphere in the feature space. See, for illustration, Figure 3.2(d) and 3.2(e) where the passage from the fourth to the fifth plot corresponds to this projection. If the features are plotted into the sample space, instead, this step corresponds to a rescaling of the axes.

A5. ℓ_1-minimization: $\min_{\hat{\mathbf{F}} \in \mathbb{R}^{L \times N}} \sum_{ij} \hat{f}_{i,j}$. This minimization is the objective of SF; by minimizing

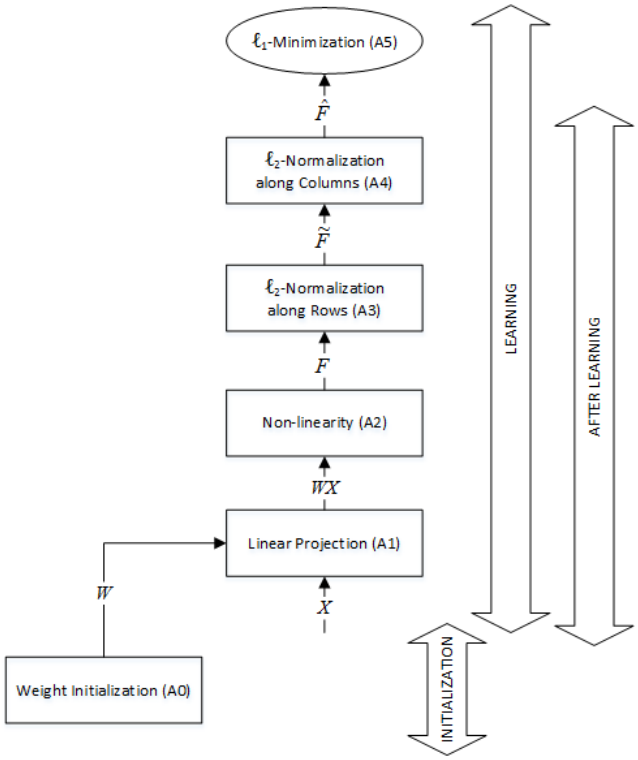

Figure 3.1: Flow chart of the SF algorithm.

the overall activation of the matrix $\hat{\mathbf{F}}$, the sparsity of the learned representations is maximized. As explained by Ngiam et al. (2011), the combination of the ℓ_1-minimization with the two ℓ_2-normalizations guarantees the learning of representations with the properties of population sparsity, lifetime sparsity and high dispersal.

These steps of the SF algorithms are executed differently in successive phases of learning and deployment. During the initialization of the algorithm, only the step A0 is executed. During training, all the steps from A1 to A5 are executed. The problem of learning a weight matrix $\mathbf{W}$ that would lead to a learned representation matrix $\mathbf{Z}$ with maximal sparsity is cast as an optimization problem that can be solved by gradient descent. SF solves this problem by relying on standard gradient descent algorithms, such as L-BFGS (Orr and Müller, 2003). After training, when processing new data, only the steps from A1 to A4 are executed. Using the learned weight matrix $\mathbf{W}$, new data are processed through the four steps from A1 and A4, and the sparse matrix $\hat{\mathbf{F}}$ is returned as the new representation of the data.

3.2.3 Properties of sparse filtering

Before moving onto the theoretical study of SF, it is useful to comment upon a couple of simple properties of the algorithm. From the description of the algorithm offered above we can immediately derive the following properties.

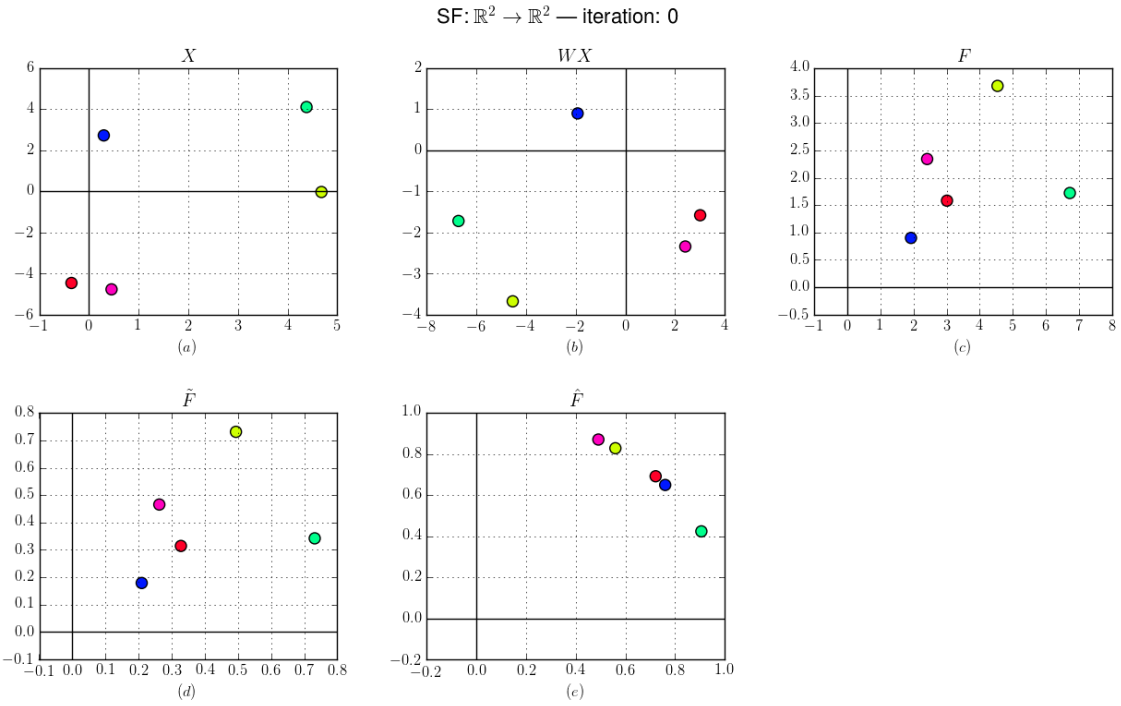

Figure 3.2: Illustration of the SF algorithm.
SF is applied to a random set of data $\mathbf{X}$ of five samples ($N = 5$) in two dimensions ($M = 2$). Each point is generated by sampling its coordinates from a uniform distribution $\mathcal{U}(-5, 5)$. SF is used to learn a new representation of the data in two dimensions ($L = 2$). This figure shows the transformations determined by the SF algorithm at iteration 0, after the weight matrix $\mathbf{W}$ has been randomly initialized and before any training.
(a) Original representation of the data $\mathbf{X}$ in $\mathbb{R}^2$. (b) Linear projection of the data onto the intermediate representation $\mathbf{WX}$. (c) Non-linear projection of $\mathbf{WX}$ using a soft absolute-value function onto the intermediate representation $\mathbf{F}$. (d) ℓ_2-normalization of the data $\mathbf{F}$ along the features, yielding the intermediate representation $\tilde{\mathbf{F}}$. (e) ℓ_2-normalization of the data $\tilde{\mathbf{F}}$ along the samples, yielding the final learned representation $\hat{\mathbf{F}} = \mathbf{Z}$. Notice that, since no learning has happened yet, the learned representations $\mathbf{Z}$ are not yet sparse.
Notice that the colours of the data points $\mathbf{x}_i$ do not have any meanings. A random colour has been assigned to each point in order to allow the tracking of the location of the points through the different transformations applied by SF.

Dependence of the representation on all the samples. In step A3 of the SF algorithm, the ℓ_2-normalization rescales the value of each feature with respect to the value of the same feature in all the other samples. This fact has important implications on the way in which SF can be used. In general, it is not possible to process an individual sample $\mathbf{x}_i$ alone, otherwise the learned representation would be trivially reduced to a vector of ones independently from the original values. It is therefore necessary to process the sample $\mathbf{x}_i$ along with other samples in the matrix $\mathbf{X}$. Given a new sample $\mathbf{x}'$ received after training, the simplest solution to learn the representation $\mathbf{z}'$ consists in processing $\mathbf{x}'$ along with the whole training data matrix $\mathbf{X}^{\mathbf{tr}}$ or the whole test data matrix $\mathbf{X}^{\mathbf{tst}}$. Beside this simple and immediate solution, more refined solutions could be conceived, and some of them are presented in the discussion in Section 5.2.3.

Positivity of the representations. A key property for the correct execution of SF is the positivity of the representations. If the intermediate representations were to assume negative values, then the dynamics of projection of the samples on the unit hypersphere in the positive orthant in step A4 would not work.

Therefore, all the steps from A2 onwards preserve the positivity of the representation. In step A2, an arbitrary non-linearity is applied to $\mathbf{WX}$. As discussed, all concrete implementations of SF rely on an absolute-value non-linearity. Now, the analytic absolute-value function $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ guarantees this property because of the non-negativity of its co-domain, while the soft absolute-value function $f : \mathbb{R} \to \mathbb{R}_{>0}$ guarantees this property thanks to the strict positivity of the co-domain. In step A3 and A4, the ℓ_2-normalizations preserve the positivity as they simply divide all the elements of $\mathbf{F}$ or $\tilde{\mathbf{F}}$ by a term given by the sum of positive terms. Finally, step A5 does not affect the representation $\mathbf{Z}$.

With this basic understanding of SF we now proceed to a more formal and thorough analysis of the algorithm in order to explain precisely its dynamics.

3.3 Theoretical analysis of sparse filtering

SF has been proved to be a successful unsupervised algorithm, but very little theoretical justification has been provided to explain its results. Here, starting from the alternative definition of distribution learning that we proposed in Section 3.1 and relying on the observations made in Section 3.2, we will deploy a set of conceptual tools, conjectures, definitions and proofs to demonstrate the following thesis:

> *SF does satisfy the informativeness principle through the maximization of the proxy of sparsity and it satisfies the infomax principle through the constraint of preservation of the structure of cosine neighbourhoodness of the data.*

Section 3.3.1 shows how SF satisfies the informativeness principle, while Section 3.3.2 introduces the hypothesis that SF satisfies the infomax principle through the preservation of the structure of the data. The hypothesis on the preservation of structure is then analysed in details in the following sections: Section 3.3.2.1 rules out the simplest hypothesis that SF preserves

a structure explained by the Euclidean metric; Section 3.3.2.2 proves that SF preserves collinearity; Section 3.3.2.3 proves that collinear points are mapped onto identical representations; similarly, Section 3.3.2.4 proves that points having the same moduli are mapped onto identical representations; and, finally, Section 3.3.2.5 puts together these results to conclude that SF preserves relations of cosine neighbourhoodness. Section 3.3.3 and Section 3.3.4 delve deeper into the dynamics of SF by providing a geometric interpretation of the algorithm in terms of bases of the learned space and filters in the original space. These concepts are then used in Section 3.3.5 to draw a consistent comparison with other sparse learning algorithms. Section 3.3.6 and Section 3.3.7 investigate the limits of the SF algorithm by evaluating the role of the absolute-value non-linearity in the preservation of structure and by deriving a probabilistic bound on the preservation of Euclidean structure. Section 3.3.8 then brings together all the results by discussing the use of SF as a representation learning algorithm. Finally, Section 3.3.9 summarizes this study by drawing a conclusion in reference to the thesis that we stated above.

3.3.1 Informativeness principle

Showing that SF satisfies the informativeness principle is straightforward. Since the explicit minimization of entropy $H_S[Z]$ is computationally hard, the SF algorithm adopts the standard proxy of sparsity. Increasing the sparsity of the representations $\mathbf{z}_i$ concentrates the mass of the pdf $p(Z)$ around zero; as the pdf $p(Z)$ gets closer to a Dirac delta function, its entropy is $H_S[Z]$ is minimized (Principe, 2010; Hurley and Rickard, 2009). Using the formalism of Pastor et al. (2015):

$$-\ell_1(\mathbf{Z}) \uparrow \quad \equiv \quad H_S[Z] \downarrow,$$

that is, as the sparsity, measured by the negative ℓ_1-norm of the learned representations $\mathbf{z}_i$, increases, so the entropy of the pdf $p(Z)$ decreases. This allows us to assert that the maximization of sparsity may work as a proxy for the minimization of entropy.

3.3.2 Infomax principle

Showing that SF satisfies the informativeness principle is more challenging. By definition, as a FDL algorithm, SF does not address the problem of modelling the data distribution. However, we conjecture that, by virtue of the fact that SF works and its learned representations $\mathbf{z}_i$ allow the achievement of state-of-the-art performance when learning $p(Y|Z)$, it *must* be that the algorithm preserves information contained in the original representations $\mathbf{x}_i$. *If it were not so*, SF could simply solve its optimization problem by mapping the original data matrix $\mathbf{X}$ onto a pre-computed sparse representation matrix $\bar{\mathbf{Z}}$, containing a constant 1-sparse learned representation $\bar{\mathbf{z}}_i$, with a minimal computational complexity of $\mathcal{O}(1)$. The matrix $\bar{\mathbf{Z}}$ would have maximal sparsity, and the associated pdf $p(Z)$ would be a Dirac delta function centred on $\bar{\mathbf{z}}_i$ with minimal entropy. However, if we were to use $\bar{\mathbf{Z}}$ to perform further supervised learning with respect to a vector of label $\mathbf{Y}$, the pre-computed learned representations $\mathbf{z}_i = \bar{\mathbf{z}}_i$ would be useless as they would provide no information about the labels because of the independence between the pre-computed representations and the given labels: $p(Y|Z) = p(Y)$.

Since SF does not try to explicitly model the distribution of the original data we hypothesize that it must implicitly preserve information about the pdf $p(X)$. We hypothesize that SF preserves the information conveyed by the pdf $p(X)$ through the proxy of the preservation of data structure. The geometric structure of the data in the original space $\mathbb{R}^M$ constitutes a set of realizations of the random variable X through which we can estimate the pdf $p(X)$. Preserving relationships of neighbourhoodness (under a given metric) allows us to preserve information conveyed by the pdf $p(X)$: regions of high density and low density in the domain of $p(X)$ can be maintained by preserving relationships of neighbourhoodness in the domain of $p(Z)$. Thus, preservation of the geometric structure under a chosen metric may act as a proxy for the maximization of mutual information $MI[X;Z]$.

3.3.2.1 Non-preservation of Euclidean distances

The preservation of absolute or relative distances under the Euclidean metric is the most common way to preserve the structure of the data. However, it can be easily ruled out that SF preserves this type of structure.

Proposition 1. *Let* $\mathbf{X}$ *be the matrix of points in the original space* $\mathbb{R}^M$*. Then, the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric.*

Proof. This proposition is proved by counterexample, that is by showing that there is at least a case for which the transformations from A1 to A4 do not preserve the Euclidean distance.

Let us consider the case in which $\mathbf{x}_1$ is a vector such that $x_{1,j} = \frac{1}{\sqrt{2}}$, $\forall j$, $1 \leq j \leq M$, $\mathbf{x}_2$ is another vector such that $\mathbf{x}_2 = -\mathbf{x}_1$, $L = M$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is:

$$D_E[\mathbf{x}_1, \mathbf{x}_2] = \sqrt{\sum_{j=1}^{M} \left(\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \right)^2} = \sqrt{2M}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{aligned}
f_{A1}(\mathbf{x}_1) &= \mathbf{I}\mathbf{x}_1 = \mathbf{x}_1 & f_{A1}(\mathbf{x}_2) &= \mathbf{I}\mathbf{x}_2 = \mathbf{x}_2 \\
f_{A2}(\mathbf{x}_1) &= |\mathbf{x}_1| = \mathbf{x}_1 & f_{A2}(\mathbf{x}_2) &= |\mathbf{x}_2| = \mathbf{x}_1 \\
f_{A3}(\mathbf{x}_1) &= \left[\tfrac{x_{1,j}}{1}\right] = \mathbf{x}_1 & f_{A3}(\mathbf{x}_1) &= \left[\tfrac{x_{1,j}}{1}\right] = \mathbf{x}_1 \\
f_{A4}(\mathbf{x}_1) &= \left[\frac{x_{1,j}}{\sqrt{\sum_j x_{1,j}^2}}\right] = \mathbf{z}_1 & f_{A4}(\mathbf{x}_1) &= \left[\frac{x_{1,j}}{\sqrt{\sum_j x_{1,j}^2}}\right] = \mathbf{z}_1.
\end{aligned}$$

Thus, $f_{A1:A4}(\mathbf{x}_1) = \mathbf{z}_1$ and $f_{A1:A4}(\mathbf{x}_2) = \mathbf{z}_1$. Now, the Euclidean distance between the vectors $f_{A1:A4}(\mathbf{x}_1)$ and $f_{A1:A4}(\mathbf{x}_2)$ is:

$$D_E[\mathbf{z}_1, \mathbf{z}_1] = 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. ■

In an analogous way, it can be proved the transformations from A1 to A4 do not preserve relative Euclidean distances.

3.3.2.2 Preservation of collinearity

Having ascertained that SF cannot preserve the data structure defined by the Euclidean metric, we investigate other properties of the algorithm that may lead us to discover the preservation of alternative data structures. A first relevant observation is that SF preserves collinearity.

Theorem 1. *Let $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^M$ be collinear points in the original space $\mathbb{R}^M$. Then, the outputs of transformations from A1 to A4, that is $f_{A1:A4}(\mathbf{x}_1)$, $f_{A1:A4}(\mathbf{x}_2)$, are collinear.*

Before proving this theorem, we present a set of auxiliary lemmas.

Lemma 1. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^M$, two generic collinear vectors, and let $f : \mathbb{R}^M \to \mathbb{R}^L$ be a linear transformation defined as $f(\mathbf{u}) = \mathbf{A}\mathbf{u}$, where $\mathbf{A}$ is the matrix associated with the linear transformation. Then, $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the linear transformation f encoded by matrix $\mathbf{A}$ and let us apply it to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = \mathbf{A}\mathbf{v} = \mathbf{A}(k\mathbf{u}) = k(\mathbf{A}\mathbf{u}) = k \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

Lemma 2. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, two generic collinear vectors, and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the element-wise absolute-value function $f(\mathbf{u}) = |\mathbf{u}| = [|u_j|]$. Then $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the element-wise absolute-value function f and let us apply it to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = |\mathbf{v}| = |k\mathbf{u}| = |k| \cdot |\mathbf{u}| = |k| \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

Lemma 3. *Let us consider $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$, two collinear vectors whose components are all strictly positive, and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the ℓ_2-normalization along the features. Then $f(\mathbf{u}), f(\mathbf{v}) \in \mathbb{R}^L$ are also collinear.*

Proof. Let us consider the two collinear vectors $\mathbf{u}$ and $\mathbf{v}$. By definition, collinearity means that there exists $k \in \mathbb{R}$, $k \neq 0$, such that $\mathbf{v} = k\mathbf{u}$. Let us now consider the function of normalization along the features $f(\mathbf{u}) = \left[\frac{u_j}{\sqrt{\sum_{\mathbf{w} \in \{\mathbf{u}, \mathbf{v}, \ldots\}} w_j^2}}\right]$, where $\{\mathbf{u}, \mathbf{v}, \ldots\}$ is the set of all available vectors for normalization. Normalizing along the features means dividing each component u_j by a constant c_j equal to the ℓ_2-norm of the component j across all the available vectors, that is $f(\mathbf{u}) = \left[\frac{u_j}{c_j}\right] = \mathbf{c} \odot \mathbf{u}$, where $\mathbf{c}$ is the vector containing all the constants c_j and

$\odot$ is the element-wise product. Let us now apply the normalization along the features to the vector $\mathbf{v}$:

$$f(\mathbf{v}) = \mathbf{c} \odot \mathbf{v} = \mathbf{c} \odot (k\mathbf{u}) = k \cdot (\mathbf{c} \odot \mathbf{u}) = k \cdot f(\mathbf{u}).$$

Therefore, collinearity is preserved. ■

Lemma 4. *Let us consider $\mathbf{u} \in \mathbb{R}^L$, a vector whose components are all strictly positive, and let $f : \mathbb{R}^L \to \mathbb{R}^L$ be the ℓ_2-normalization along the samples. Then $f(\mathbf{u}) \in \mathbb{R}^L$ has the same angular coordinates as $\mathbf{u}$.*

Proof. Let us consider the function of normalization along the features $f(\mathbf{u}) = \left[\frac{u_j}{\sqrt{\sum_j u_j^2}}\right]$. Normalizing along the samples means dividing each component u_j by the ℓ_2-norm of the same vector $\mathbf{u}$, that is $f(\mathbf{u}) = \left[\frac{u_j}{\ell_2(\mathbf{u})}\right] = \frac{1}{\ell_2(\mathbf{u})}$. Now, multiplying all the components of the same vector $\mathbf{u}$ by the constant $c = \frac{1}{\ell_2(\mathbf{u})}$ leaves the angular coordinates unaltered. Therefore, the angular coordinates are preserved. ■

Proof of Theorem 1. To prove that the transformations from A1 to A4 preserve collinearity it is necessary to prove that all transformations preserve collinearity.

Concerning transformation A1, by Lemma 1, linear transformations preserve collinearity. Concerning transformation A2, by Lemma 2, the absolute-value function preserves collinearity; indeed, it rigidly folds all the orthants on the first one (see Section 3.2.2). Concerning transformation A3, by Lemma 3, normalization along the features preserves collinearity; indeed, it acts simply as a rescaling of the axes (see Section 3.2.2). Concerning transformation A4, by Lemma 4, normalization along the samples preserves angular coordinates in general, and, therefore, collinearity.

Since all the transformations from A1 to A4 preserve collinearity, the overall transformation $f_{A1:A4}$ preserves collinearity. ■

3.3.2.3 Homo-representation of collinear points

An immediate consequence of the previous result is the following theorem which states that all the collinear points in the original representation space are mapped onto an identical representation. This result is significant as it gives us a first understanding of the principle and the type of metric that SF uses to map original samples $\mathbf{x}_i$ onto their representations $\mathbf{z}_i$.

Theorem 2. *Let $\mathbf{x}_1 \in \mathbb{R}^M$ be a point in the the original space $\mathbb{R}^M$. Then there is an infinite set of points $\mathbf{x}_i \in \mathbb{R}^M$ such that $f_{A1:A4}(\mathbf{x}_1) = f_{A1:A4}(\mathbf{x}_i)$. The set of the points collinear with $\mathbf{x}_1$ is included in this set.*

Proof. Let us consider a point $\mathbf{x}_1$ and a generic collinear point $\mathbf{x}_2 = k\mathbf{x}_1$, $k \neq 0$. Let us apply the transformation $f_{A1:A4}$ to the points $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{array}{cc}
f_{A1}(\mathbf{x}_1) = \mathbf{W}\mathbf{x}_1 & f_{A1}(\mathbf{x}_2) = k\mathbf{W}\mathbf{x}_1 \\
f_{A2}(\mathbf{W}\mathbf{x}_1) = |\mathbf{W}\mathbf{x}_1| & f_{A2}(k\mathbf{W}\mathbf{x}_1) = k|\mathbf{W}\mathbf{x}_1| \\
f_{A3}(|\mathbf{W}\mathbf{x}_1|) = \mathbf{c} \odot |\mathbf{W}\mathbf{x}_1| & f_{A3}(k|\mathbf{W}\mathbf{x}_1|) = k(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|) \\
f_{A4}(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|) = \frac{\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|}{\ell_2(\mathbf{x}_1)} & f_{A4}(k(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|)) = \frac{k(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|)}{\ell_2(\mathbf{x}_2)},
\end{array}$$

where $\mathbf{c}$ is a vector of normalizing constants and $\odot$ is the element-wise product (as in Lemma 3). Now, since $\ell_2(\mathbf{x}_2) = k\ell_2(\mathbf{x}_1)$, it follows:

$$\frac{k(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|)}{\ell_2(\mathbf{x}_2)} = \frac{k(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|)}{k\ell_2(\mathbf{x}_1)} = f_{A4}(\mathbf{c} \odot |\mathbf{W}\mathbf{x}_1|).$$

Thus, $\mathbf{x}_1$ and any collinear point $\mathbf{x}_2$ are mapped onto the same representation $f_{A1:A4}(\mathbf{x}_1)$. ■

3.3.2.4 Homo-representation of points with same moduli

A further analysis of SF reveals that not only collinear points are mapped to the same representation, but also points in the learned representation space having the same moduli (that is, the same absolute value for their components) are mapped onto identical representations. Again, this result is relevant since it sheds light on the type of structure preserved by SF.

Theorem 3. *Let $\mathbf{f}_1 \in \mathbb{R}^L$ be a point in the co-domain of the linear map defined by the matrix $\mathbf{W}$. It holds that for $\mathbf{f}_1$ strictly in the first orthant, there are at least 2^L points $\mathbf{f}_i \in \mathbb{R}^L$ such that $f_{A2:A4}(\mathbf{f}_1) = f_{A2:A4}(\mathbf{f}_i)$.*

Proof. By definition, $f_{1,j} > 0$, $\forall j$, $1 \leq j \leq L$. It follows that $f_{A2}(\mathbf{f}_1) = \mathbf{f}_1$, as the application of the absolute-value maps $\mathbf{f}_1$ to itself.

However, all the vectors $\mathbf{f}_i$ such that $f_{i,j} = \pm f_{1,j}$ are mapped onto $\mathbf{f}_1$ by the absolute-value f_{A2}. By combinatorial analysis, there are 2^L possible ways of picking the values of $\mathbf{f}_i$, thus defining 2^L points in $\mathbb{R}^L$ that are mapped to the same value $\mathbf{f}_1$. Since all the points $\mathbf{f}_i$ are mapped to the same point $\mathbf{f}_1$ at the end of step A2, the application of the remaining deterministic functions will map them to the same representation, $f_{A2:A4}(\mathbf{f}_1) = f_{A2:A4}(\mathbf{f}_i)$. ■

3.3.2.5 Preservation of cosine neighbourhoodness

In Theorem 2 we have shown that SF maps points having the same angles to the same representations. However, this property is not sufficient to preserve any complex structure. Here we further prove that SF maps points having a small cosine distance $D_C[\mathbf{x}_1, \mathbf{x}_2] = 1 - \frac{\mathbf{x}_1\mathbf{x}_2}{\ell_2(\mathbf{x}_1)\ell_2(\mathbf{x}_2)}$ in the original space onto points having small Euclidean distance $D_E[\mathbf{z}_1, \mathbf{z}_2]$ in the representation space.

Theorem 4. *Let $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^M$ be two original data samples and let $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^L$ be their representations computed by SF. If the cosine distance between the original samples is arbitrarily small $D_C[\mathbf{x}_1, \mathbf{x}_2] < \delta$, for $\delta > 0$, then the Euclidean distance between the computed representations is arbitrarily small $D_E[\mathbf{z}_1, \mathbf{z}_2] < \epsilon$, for $\epsilon > 0$, and $\epsilon = L \cdot \left(\frac{k + |\sqrt{2\delta - \delta^2}|}{\ell_2(\tilde{\mathbf{f}}_2)} - \frac{1}{\ell_2(\tilde{\mathbf{f}}_1)} \right)$, where k is a constant accounting for partial collinearity and $\ell_2\left(\tilde{\mathbf{f}}_i\right)$ is the ℓ_2-norm of the representations computed by SF after step A3. In the limit, it holds that $\lim_{\delta \to 0} \epsilon = 0$.*

Proof. In order to prove this theorem we adopt the following strategy: we compute the representations at each step of the computation (before SF, after steps A1 and A2, after step A3 and after step A4) and we upper bound the displacement accounting for the Euclidean distance between the representations. Lastly, we prove the behaviour of our relationship in the limit.

Recall that given two generic points $\mathbf{x}_1$ and $\mathbf{x}_2$, we can express $\mathbf{x}_2$ as a function of $\mathbf{x}_1$ plus a *displacement* vector $\bar{\mathbf{x}}$:

$$\mathbf{x}_2 = \mathbf{x}_1 + \bar{\mathbf{x}}, \tag{3.2}$$

so that we can easily account for the Euclidean distance between $\mathbf{x}_1$ and $\mathbf{x}_2$ just as a function of the displacement vector $\bar{\mathbf{x}}$:

$$D_E\left[\mathbf{x}_1, \mathbf{x}_2\right] = \ell_2\left(\bar{\mathbf{x}}\right).$$

(Before SF.) Let us now consider two points $\mathbf{x}_1$ and $\mathbf{x}_2$ which are almost collinear with an arbitrary small cosine distance $D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] < \delta$. We can then express $\mathbf{x}_1$ as a point collinear with $\mathbf{x}_2$ to which a *bias* vector $\mathbf{b}$ is added:

$$\mathbf{x}_2 = k\mathbf{x}_1 + \mathbf{b},$$

where $k \in \mathbb{R}$, $k \neq 0$ is a constant that preserves collinearity. With no loss of generality, we will assume $k > 1$; we exclude values of k smaller than zero which would generate a reflection (reflections are not relevant for the following treatment as they induce a cosine distance far greater than δ) and we ignore values of k falling between zero and one (in such a case, the proof will hold once we swap $\mathbf{x}_1$ and $\mathbf{x}_2$). The bias vector $\mathbf{b}$ accounts for a relative displacement between the perfectly collinear sample $k\mathbf{x}_1$ and the almost collinear sample $k\mathbf{x}_1 + \mathbf{b}$.

With reference to Equation 3.2, the displacement vector $\bar{\mathbf{x}}$ is:

$$\bar{\mathbf{x}} = (k-1)\mathbf{x}_1 + \mathbf{b}, \tag{3.3}$$

from which follows that:

$$D_E\left[\mathbf{x}_1, \mathbf{x}_2\right] = \ell_2\left(\bar{\mathbf{x}}\right) = \ell_2\left((k-1)\mathbf{x}_1 + \mathbf{b}\right).$$

(Before SF - Upper bound) To upper bound $D_E\left[\mathbf{x}_1, \mathbf{x}_2\right]$, we can evaluate the maximum value that $\ell_2\left(\bar{\mathbf{x}}\right)$ can reach, consistent with the constraint of a bounded cosine distance $D_C\left[\mathbf{x}_1, \mathbf{x}_2\right]$. Formally, we set up the optimization problem:

$$\underset{\bar{\mathbf{x}} \in \mathbb{R}^M}{\operatorname{argmax}}\, \ell_2\left(\bar{\mathbf{x}}\right),$$

under the constraint:

$$D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] < \delta.$$

The maximization can be rewritten as:

$$\begin{aligned}
\underset{\bar{\mathbf{x}}\in\mathbb{R}^M}{\operatorname{argmax}}\, \ell_2\left(\bar{\mathbf{x}}\right) &= \underset{\bar{x}_j\in\mathbb{R}}{\operatorname{argmax}} \sqrt{\sum_{j=1}^{M} \bar{x}_j^2} \\
&= \underset{b_j\in\mathbb{R}}{\operatorname{argmax}} \sqrt{\sum_{j=1}^{M} \left((k-1)x_{1,j}+b_j\right)^2} \\
&= \underset{b_j\in\mathbb{R}}{\operatorname{argmax}} \sqrt{\sum_{j=1}^{M} b_j^2} \\
&= \underset{b_j\in\mathbb{R}}{\operatorname{argmax}}\, b_j,
\end{aligned}$$

assuming: (i) that $\mathbf{x}_1$ and k are given and fixed, and (ii) that $x_{1,j}$ and b_j are both positive (as this constitutes the worst case that needs to be considered in the analysis of the upper bound). An upper bound on the displacement $\bar{\mathbf{x}}$ can be then computed from the solution to the individual constrained optimization problems for each component b_j:

$$\max_{b_j\in\mathbb{R}} b_j,$$

under the constraint:

$$\begin{aligned}
\delta &> D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] \\
&= D_C\left[\mathbf{x}_1, k\mathbf{x}_1 + \mathbf{b}\right].
\end{aligned}$$

By construction, we know that $D_C\left[\mathbf{x}_1, k\mathbf{x}_1\right] = 0$. Therefore the entire cosine distance must be accounted by the bias vector $\mathbf{b}$. Trigonometrically, from the cosine distance δ we can recover the angle opposite to a cathetus corresponding to the radius of an hypersphere centred on $k\mathbf{x}_1$ and bounding the module of $\mathbf{b}$. Let θ be the underlying angle between $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{aligned}
\delta &= 1 - \cos\left(\theta\right) \\
\theta &= \arccos\left(1-\delta\right).
\end{aligned}$$

The radius of the hypersphere centred on $k\mathbf{x}_1$ inducing at most a cosine distance δ is:

$$\begin{aligned}
b_j &\leq x_{1,j} \sin\left(\arccos\left(1-\delta\right)\right) \\
&= x_{1,j}\sqrt{1-(1-\delta)^2} \\
&= x_{1,j}\sqrt{2\delta-\delta^2}.
\end{aligned}$$

Substituting this value in Equation 3.3, the displacement on each component can the be upper

bounded as:

$$\begin{aligned}\bar{x}_j &= (k-1)x_{1,j} + b_j \\ &\leq (k-1)x_{1,j} + x_{1,j}\sqrt{2\delta - \delta^2} \\ &= x_{1,j}\left(k - 1 + \sqrt{2\delta - \delta^2}\right).\end{aligned}$$

This upper bound depends on the original cosine distance δ, but more significantly on the module of $\mathbf{x}_1$ and the stretching constant k. Indeed, the Euclidean distance along each component is given by the stretch $(x_{1,j}(k-1))$ plus a small distance due to the angle $(x_{1,j}\sqrt{2\delta - \delta^2})$.

(Steps A1 and A2) Let us now apply the linear projection and the absolute-value function defined in transformation A1 and A2:

$$\begin{aligned}\mathbf{f}_1 = f_{A1:A2}(\mathbf{x}_1) &= |\mathbf{W}\mathbf{x}_1| \\ \mathbf{f}_2 = f_{A1:A2}(\mathbf{x}_2) &= |\mathbf{W}(k\mathbf{x}_1 + \mathbf{b})| = k\mathbf{f}_1 \pm |\mathbf{W}\mathbf{b}|.\end{aligned}$$

Component-wise we have:

$$\begin{aligned}f_{1,l} &= \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| \\ f_{2,l} &= kf_{1,l} + |\mathbf{W}\mathbf{b}|_l = k\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| \pm \left|\sum_{j=1}^{M} w_{j,l}b_j\right|.\end{aligned}$$

The new displacement and the new Euclidean distance are:

$$\bar{f}_l = (k-1)f_{1,l} \pm |\mathbf{W}\mathbf{b}|_l \tag{3.4}$$

$$D_E[\mathbf{f}_1, \mathbf{f}_2] = \ell_2(\bar{\mathbf{f}}) = \sqrt{\sum_{l=1}^{L}((k-1)f_{1,l} \pm |\mathbf{W}\mathbf{b}|_l)^2}.$$

(Steps A1 and A2 - Upper bound) The upper bound of each component of the new bias vector follows immediately:

$$\begin{aligned}|\mathbf{W}\mathbf{b}|_l &= \left|\sum_{j=1}^{M} w_{j,l}b_j\right| \\ &\leq \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\sqrt{2\delta - \delta^2}\right| \\ &= \left|\sqrt{2\delta - \delta^2}\right|\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|,\end{aligned}$$

and then the upper bound on each component of the displacement in Equation 3.4 is:

$$\begin{aligned}\bar{f}_l &\leq (k-1)f_{1,l} + \left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| \\ &= (k-1)\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| + \left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| \\ &= \left(k-1+\left|\sqrt{2\delta-\delta^2}\right|\right) \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|.\end{aligned}$$

(Step A3) Let us now apply the normalization along the rows defined in transformation A3:

$$\begin{aligned}\tilde{f}_{1,l} = f_{A3}\left(f_{1,l}\right) &= \frac{f_{1,l}}{\sqrt{\sum_{i=1}^{N} f_{i,l}^2}} \\ \tilde{f}_{2,l} = f_{A3}\left(f_{2,l}\right) &= \frac{kf_{1,l} + |\mathbf{Wb}|_l}{\sqrt{\sum_{i=1}^{N} f_{i,l}^2}} = k\tilde{f}_{1,l} + \frac{|\mathbf{Wb}|_l}{\sqrt{\sum_{i=1}^{N} f_{i,l}^2}}.\end{aligned}$$

Notice that the denominator is given by a feature-dependent sum across N samples; for simplicity, we will take this value to be a constant $\{c_l\}_{l=1}^{L}$, $c_l \in \mathbb{R}$:

$$\begin{aligned}\tilde{f}_{1,l} &= \frac{f_{1,l}}{c_l} \\ \tilde{f}_{2,l} &= k\tilde{f}_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l}.\end{aligned}$$

The new displacement and the new Euclidean distance are:

$$\bar{\tilde{f}}_l = (k-1)\tilde{f}_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l} \tag{3.5}$$

$$D_E\left[\tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2\right] = \ell_2\left(\bar{\tilde{\mathbf{f}}}\right) = \sqrt{\sum_{l=1}^{L}\left((k-1)\tilde{f}_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l}\right)^2}.$$

(Step A3 - Upper bound) The upper bound of each component of the new bias vector follows immediately:

$$\frac{|\mathbf{Wb}|_l}{c_l} \leq \frac{\left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|}{c_l},$$

and then the upper bound on each component of the displacement in Equation 3.5 is:

$$\begin{aligned}
\bar{\tilde{f}}_l &\leq (k-1)\tilde{f}_{1,l} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|}{c_l} \\
&= (k-1)\frac{f_{1,l}}{c_l} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|}{c_l} \\
&= (k-1)\frac{\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|}{c_l} + \frac{\left|\sqrt{2\delta-\delta^2}\right|\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right|}{c_l} \\
&= \frac{k-1+\left|\sqrt{2\delta-\delta^2}\right|}{c_l}\left|\sum_{j=1}^{M} w_{j,l}x_{1,j}\right| \\
&= \frac{1}{c_l}\bar{f}_l.
\end{aligned}$$

Not surprisingly, after transformation A3, the Euclidean distance $D_E\left[\tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2\right]$ is just rescaled since each component of the displacement $\bar{f}_l$ is reduced by a factor $\frac{1}{c_l} = \frac{1}{\sqrt{\sum_{i=1}^{N} f_{i,l}^2}}$.

(Step A4) Finally, let us apply the normalization along the samples defined in transformation A4:

$$\begin{aligned}
z_{1,l} &= f_{A4}\left(\tilde{f}_{1,l}\right) = \frac{\tilde{f}_{1,l}}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} = \frac{\frac{f_{1,l}}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} \\
z_{2,l} &= f_{A4}\left(\tilde{f}_{2,l}\right) = \frac{\tilde{f}_{2,l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} = \frac{k\tilde{f}_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} = \frac{k\frac{f_{1,l}}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} + \frac{\frac{|\mathbf{Wb}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)}.
\end{aligned}$$

Let us now consider the first term of $z_{2,l}$ and let us multiply it by $\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}$:

$$z_{2,l} = \frac{k\frac{f_{1,l}}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} \cdot \frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} + \frac{\frac{|\mathbf{Wb}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} = kz_{1,l}\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} + \frac{\frac{|\mathbf{Wb}|_l}{c_l}}{\ell_2\left(\tilde{\mathbf{f}}_2\right)}.$$

The new displacement and the new Euclidean distance are:

$$\bar{z}_l = \left(k\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - 1\right) z_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l\ell_2\left(\tilde{\mathbf{f}}_2\right)} \tag{3.6}$$

$$D_E\left[\mathbf{z}_1, \mathbf{z}_2\right] = \ell_2\left(\bar{\mathbf{z}}\right) = \sqrt{\sum_{l=1}^{L}\left(\left(k\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - 1\right) z_{1,l} + \frac{|\mathbf{Wb}|_l}{c_l\ell_2\left(\tilde{\mathbf{f}}_2\right)}\right)^2}. \tag{3.7}$$

For consistency, notice that if $\mathbf{x}_1$ and $\mathbf{x}_2$ were to be collinear, then $\ell_2\left(\tilde{\mathbf{f}}_2\right) = k\ell_2\left(\tilde{\mathbf{f}}_1\right)$, and, by construction, $\mathbf{b} = 0$; therefore, in case of collinearity, $D_E\left[\mathbf{z}_1, \mathbf{z}_2\right]$ computed in Equation 3.7 would be zero, thus agreeing with Theorem 2.

(Step A4 - Upper bound) Now, the upper bound of each component of the bias vector can be immediately evaluated:

$$\frac{|\mathbf{Wb}|_l}{c_l \ell_2\left(\tilde{\mathbf{f}}_2\right)} \leq \frac{\left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l \ell_2\left(\tilde{\mathbf{f}}_2\right)},$$

and then the upper bound on each component of the displacement:

$$\begin{aligned}
\bar{z}_l &\leq \left(k\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)}-1\right) z_{1,l} + \frac{\left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l \ell_2\left(\tilde{\mathbf{f}}_2\right)} \\
&= \left(k\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)}-1\right) \frac{f_{1,l}}{c_l \ell_2\left(\tilde{\mathbf{f}}_1\right)} + \frac{\left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l \ell_2\left(\tilde{\mathbf{f}}_2\right)} \\
&= \left(k\frac{\ell_2\left(\tilde{\mathbf{f}}_1\right)}{\ell_2\left(\tilde{\mathbf{f}}_2\right)}-1\right) \frac{\left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l \ell_2\left(\tilde{\mathbf{f}}_1\right)} + \frac{\left|\sqrt{2\delta-\delta^2}\right| \left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l \ell_2\left(\tilde{\mathbf{f}}_2\right)} \\
&= \frac{\left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l} \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right).
\end{aligned}$$

Notice that $\frac{\left|\sum_{j=1}^{M} w_{j,l} x_{1,j}\right|}{c_l} < 1$ since $c_l = \sqrt{\sum_{i=1}^{N} f_{i,l}^2}$. Thus:

$$\bar{z}_l \leq \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right).$$

The overall Euclidean distance between the representations $\mathbf{z}_1$ and $\mathbf{z}_2$ can then be bounded by:

$$\begin{aligned}
D_E\left[\mathbf{z}_1, \mathbf{z}_2\right] &= \sqrt{\sum_{l=1}^{L} \bar{z}_l^2} \\
&\leq L \cdot \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right).
\end{aligned}$$

Thus $\epsilon = L \cdot \left(\frac{k+\left|\sqrt{2\delta-\delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)}\right)$.

(Limit case) Lastly, let us consider the behaviour of the Euclidean distance $D_E\left[\mathbf{z}_1, \mathbf{z}_2\right]$ as

the the cosine distance $D_C[\mathbf{x}_1, \mathbf{x}_2]$ tends to zero:

$$\begin{aligned}\lim_{\delta\to 0} D_E[\mathbf{z}_1, \mathbf{z}_2] &= \lim_{\delta\to 0} \epsilon \\ &= \lim_{\delta\to 0} L \cdot \left(\frac{k + \left|\sqrt{2\delta - \delta^2}\right|}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} \right) \\ &= \lim_{\delta\to 0} L \cdot \left(\frac{k}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} \right).\end{aligned}$$

Let us now substitute $\ell_2\left(\tilde{\mathbf{f}}_2\right)$ with its definition. As the cosine distance δ tends to zero, $\mathbf{x}_1$ and $\mathbf{x}_2$ tend to be collinear. Therefore, $\ell_2\left(\tilde{\mathbf{f}}_2\right)$ tends to $k\ell_2\left(\tilde{\mathbf{f}}_1\right)$. We can then rewrite:

$$\begin{aligned}\lim_{\delta\to 0} D_E[\mathbf{z}_1, \mathbf{z}_2] &= \lim_{\delta\to 0} L \cdot \left(\frac{k}{\ell_2\left(\tilde{\mathbf{f}}_2\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} \right) \\ &= \lim_{\delta\to 0} L \cdot \left(\frac{k}{k \cdot \ell_2\left(\tilde{\mathbf{f}}_1\right)} - \frac{1}{\ell_2\left(\tilde{\mathbf{f}}_1\right)} \right) \\ &= 0.\end{aligned}$$

Thus, in the limit, it holds that $\lim_{\delta\to 0} \epsilon = 0$. ■

SF can then preserve cosine neighbourhoodness by mapping points that have similar angular coordinates onto representations that are close to each other under Euclidean distance. In particular, as the cosine distance δ in the original space tends to zero, the Euclidean distance ϵ consistently tends to zero, $\lim_{\delta\to 0} \epsilon = 0$.

However, notice that points that have a large cosine distance in the original space will not necessarily be far in the representation space; this is a consequence of the fact that transformations in SF preserve collinearity and cosine neighbourhoodness, but not cosine metric in general.

3.3.3 Basis and basis pursuit

Let us now consider the space of the learned representations $\mathbb{R}^L$. This space is spanned by the canonical set of orthonormal bases $\{\mathbf{e}_i\}_{i=1}^L$, where $\mathbf{e}_1 = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}$, $\mathbf{e}_2 = \begin{bmatrix} 0 & 1 & \dots & 0 \end{bmatrix}$, $\dots$, $\mathbf{e}_L = \begin{bmatrix} 0 & 0 & \dots & 1 \end{bmatrix}$.

Let us consider the vectors $\mathbf{z}_i$ produced by the SF algorithm through the steps A1 to A4. Considering the optimization in step A5, it is easy to prove that the optimal set $\{\mathbf{z}_i\}$ which minimizes the ℓ_1-norm is given by a multi-set[1] of the orthonormal bases of $\mathbb{R}^L$.

Proposition 2. *Let $\{\mathbf{z}_i\}$ be a set of vectors in $\mathbb{R}^L$ such that $\sum_{j=1}^L z_{i,j}^2 = 1$. Then, an optimal*

[1] We now explicitly refer to $\{\mathbf{z}_i\}$ as a multi-set because the optimal set may contain repeated orthonormal bases of $\mathbb{R}^L$ in case $N > L$.

set of vectors that solves the optimization problem $\min_{\mathbf{Z}\in\mathbb{R}^{L\times N}} \sum_{i=1}^{N}\sum_{j=1}^{L} z_{i,j}$ *is given by a multi-set of the orthonormal bases of* $\mathbb{R}^L$.

Proof. This proposition is proved geometrically, following the proof given by Bishop (2007) to show the sparsity of the solutions of the regularized least squares optimization problem.

Let us consider the optimization problem:

$$\min_{\mathbf{z}_1\in\mathbb{R}^L} \sum_{j=1}^{L} |z_{1,j}|\,,$$

subject the constraint:

$$\sum_{j=1}^{L} z_{1,j}^2 = 1.$$

The constraint defines the set of points describing a unitary hyper-sphere in $\mathbb{R}^L$, while the minimization problem defines diamond-shaped level sets (Bishop, 2007). The minimal level set intersecting the unitary hyper-sphere is the diamond inscribed in the unit sphere. The intersection points constitute the solution of the minimization problem. These points are the intersection points between the unit hyper-sphere and the axes of $\mathbb{R}^L$, having a single component set to one, while all the others are set to zero. By definition, these 1-sparse vectors are the orthonormal bases $\mathbf{e}_i$. ■

Thus, the optimal solution for the SF algorithm is to map a set of original representations $\mathbf{x}_i \in \mathbb{R}^M$ onto the orthonormal bases of $\mathbb{R}^L$, as the bases $\mathbf{e}_i$ have a minimal ℓ_1-norm in $\mathbb{R}^L$ under the constraint of SF.

Ideally, through gradient descent, SF progressively pushes the learned representations $\mathbf{z}_i \in \mathbb{R}^L$ towards the orthonormal bases of $\mathbb{R}^L$. However, in general, notice that SF is not guaranteed to find a solution in which all the original representations $\mathbf{x}_i$ are mapped onto bases $\mathbf{e}_i$. The achievement of such an optimal solution depends on the original data set $\mathbf{X}$, on the dimensionality of the learned space L and on the random initialization of the weight matrix $\mathbf{W}$. Gradient descent in a non-convex space may lead SF to settle into a local minimum, that is a sub-optimal solution where the original representations $\mathbf{x}_i$ are not mapped onto bases but onto k-sparse ($k > 1$) representations in $\mathbb{R}^L$.

3.3.4 Representation filters

Understanding the internal workings of the SF algorithm in terms of orthonormal bases and pursuit of these bases allows us to introduce a last conceptual tool that gives us a better insight into the properties and the dynamics of SF.

From Theorem 2 we learned that SF identifies sets of collinear points in the original space to be mapped onto bases; from Theorem 3 we can deduce that there must a symmetric structure around lines of collinear points; from Theorem 4 we learned that cosine neighbourhoodness is translated into Euclidean neighbourhoodness. Putting together these results, we can infer that SF defines precise maps in the original representation space $\mathbb{R}^M$. More precisely, we state that

SF defines *representation filters* in the form of *hyper-conical filters* in the original representation space $\mathbb{R}^M$.

Definition (Representation Filter). A *representation filter* $R^{\mathbf{e}_i}$ is a function $R^{\mathbf{e}_i} : \mathbb{R}^M \rightarrow \mathbb{R}_{\geq 0}$ mapping points in the original representation space $\mathbb{R}^M$ onto their Euclidean distance from the basis $\mathbf{e}_i$.

Plotting a representation filter $R^{\mathbf{e}_i}$ in the original space $\mathbb{R}^M$ defines a region of space having a hyper-conical shape, such that all the points on the line of its height are mapped onto the basis $\mathbf{e}_i$, and all the points in the neighbourhood defined by its volume are mapped into the neighbourhood of the basis $\mathbf{e}_i$. Moreover, given a point $\mathbf{x}_1 \in \mathbb{R}^M$, we say that the representation filter $R^{\mathbf{e}_i}_{\mathbf{x}_1}$ is centred on $\mathbf{x}_1$ if $R^{\mathbf{e}_i}_{\mathbf{x}_1}(\mathbf{x}_1) = 0$, that is, the point $\mathbf{x}_1$ lies on the line of the height of the representation hyper-cone defined by $R^{\mathbf{e}_i}_{\mathbf{x}_1}$.

Several useful properties immediately follow from the definition of representation filters:

- *Association with a basis:* each filter is associated with a basis of $\mathbb{R}^L$.
- *Existence of L representation filters:* SF defines exactly L filters. This statement clearly follows from the fact that in $\mathbb{R}^L$ there are exactly L orthonormal bases.
- *M-dimensionality of the representation filters:* in the original representation space $\mathbb{R}^M$, SF defines M-dimensional hyper-conical filters. If the original space is two-dimensional, the representation filters are cone-shaped; in higher dimensions the representation filters are hyper-cones.
- *Bounds of representation filters:* given a point $\mathbf{x}_i$ in the original space $\mathbb{R}^M$, then $0 \leq R^{\mathbf{e}_i}(\mathbf{x}_i) \leq \sqrt{2}$. Since each point $\mathbf{x}_i$ is mapped onto a point $\mathbf{z}_i$ on the surface of the unit hyper-sphere in the positive orthant, the distance of $\mathbf{z}_i$ from any basis of $\mathbb{R}^L$ is bounded between 0 and $\sqrt{2}$.
- *Closeness to the filters:* the representation filters comply with a rule of inverse proportionality: the closer a point $\mathbf{x}_i$ approaches a basis $\mathbf{e}_i$, the further it moves away from all other bases.
- *Complementarity of the representation filters:* inspecting a plot of the representation filters gives us a rapid intuitive idea of the quality of the solution: points on the line of height of a representation filter $R^{\mathbf{e}_i}$ are mapped onto a perfect 1-sparse representation (basis); points within the volume of a representation cone $R^{\mathbf{e}_i}$ are mapped in the neighbourhood of a basis; points far from any representation filter are mapped onto sub-optimal k-sparse representations, with $1 < k \leq L$.
- *Learning:* after initialization, the representation filters are randomly placed in the original representation space $\mathbb{R}^M$. This leads to an unsatisfactory solution, as random points potentially far from the samples $\mathbf{x}_i$ may be mapped onto bases. During learning, SF performs a pursuit of the orthonormal bases moving the representation filters so that they may be centred on samples $\mathbf{x}_i$. The optimization process of SF can be interpreted as the *search for an optimal location of the representation filters*: hyper-conical representation filters are rotated in a continuous way in the original representation space, until their

placement provides an optimal solution in terms of sparsity of the learned representations. The optimal solution to the problem of minimizing the ℓ_1-norm of the learned representations $\mathbf{z}_i$ is equivalent to the optimal solution of the problem of minimizing the distances defined by $R^{\mathbf{e}_i}$ of the original representations $\mathbf{x}_i$.

3.3.5 Sparse filtering and other sparse learning algorithms

Understanding the dynamics of sparse filtering in terms of bases and filters naturally prompts a comparison with other popular sparse learning techniques used in signal processing and machine learning. *Basis pursuit* (Chen et al., 2001) defines an optimization problem aimed at discovering a maximally sparse representation of a signal:

$$\begin{aligned} \underset{\mathbf{z}_i \in \mathbb{R}^L}{\text{argmin}} \quad & \ell_1(\mathbf{z}_i) \\ \text{subject to} \quad & \mathbf{x}_i = \mathbf{D}\mathbf{z}_i, \end{aligned}$$

where the matrix $\mathbf{D}$ is called measurement matrix and its columns are taken to be measurement bases. The formulation of this problem is close to sparse filtering. Indeed, the objective of basis pursuit is the same as sparse filtering: an ℓ_1-minimization problem, mapping the original representations $\mathbf{x}_i$ onto the bases of a space defined by a dictionary. However, the constraints in the two settings are different: in basis pursuit the constraint is given by a linear transformation defined by a given measurement matrix, while in sparse filtering the constraints are expressed through the transformations defined by the algorithm. Several methods, such as *iterative shrinkage thresholding* (Figueiredo and Nowak, 2003) and *alternative direction method* (Yang and Zhang, 2011), have been proposed to solve the basis pursuit problem (Zhang et al., 2015). The problem of learning a sparse decomposition of a signal may also be solved using greedy algorithms, such as the *matching pursuit* (Mallat and Zhang, 1993) algorithm. Matching pursuit aims at solving the following optimization problem:

$$\begin{aligned} \underset{\mathbf{d}_j \in \mathcal{D}}{\text{argmin}} \quad & R\left(\mathbf{x}_i, \sum_j \mathbf{d}_j z_{i,j}\right) \\ \text{subject to} \quad & \ell_0(\mathbf{D}) < T, \end{aligned}$$

where $\mathbf{d}_j \in \mathcal{D}$ is a basis or an atom taken from a dictionary of bases $\mathcal{D}$, $R(\cdot)$ is a residual measure (such as the squared norm) used to compute the difference between the original representation and the learned representation and $T \in \mathbb{N}$ is a threshold. A greedy iterative technique is used to select a minimal set of bases $\mathbf{d}_j$, thus generating a sparse solution $\mathbf{z}_i$. Despite the common aim of learning a sparse decomposition of the original representations, significant differences exist between the two algorithms: matching pursuit looks for a linear decomposition, while sparse filtering considers a non-linear decomposition; matching pursuit selects one basis at each iteration from a given set of bases, while sparse filtering optimizes at each iteration all the bases. Other variations and optimizations of the basic matching pursuit algorithm, such as *orthogonal matching pursuit* (Pati et al., 1993), *compressive sampling matching pursuit* (Needell and Tropp, 2009) or *kernel matching pursuit* (Vincent and Bengio, 2002), relate to sparse filtering in the

same way. A connection may be established with dictionary learning algorithms (Rubinstein et al., 2010) as well, such as the *method of optimal directions* (Engan et al., 1999) or *k-singular value decomposition* (Aharon et al., 2006). These algorithms aim to find a solution to the following optimization problem:

$$\begin{aligned} \underset{\mathbf{D}\in\mathbb{R}^{L\times M},\mathbf{Z}in\mathbb{R}^{L}}{\text{argmin}} \quad & R\left(\mathbf{X},\mathbf{DZ}\right) \\ \text{subject to} \quad & \ell_0\left(\mathbf{D}\right) < T. \end{aligned}$$

Similarly to sparse filtering, dictionary learning algorithms try to learn a dictionary and a sparse representation at the same time. However, while dictionary learning algorithms typically alternate between updating the dictionary and the sparse representation, sparse filtering explicitly optimizes only the sparsity of the learned representation.

3.3.6 Non-preservation of cosine neighbourhoodness in alternative implementations of sparse filtering

The choice of the non-linearity applied in step A2 of SF is crucial for guaranteeing the preservation of cosine neighbourhoodness. Indeed, we argue that the absolute-value non-linearity is a suitable non-linear function for SF precisely because it preserves a relevant structure.

Ngiam et al. (2011) suggest that the original absolute-value non-linearity may be substituted by other non-linear functions; for instance, it may be possible to swap the absolute-value function for standard non-linear functions from the neural networks literature, such as the sigmoid non-linearity or the rectified linear unit (ReLU) (Nair and Hinton, 2010). Despite this possibility, all the successful implementations of SF so far have relied on the absolute-value non-linearity. An unpublished technical report by Thaler[2] states that SF with alternative non-linearities (ReLU and quadratic non-linearity) does not perform as well as with the absolute-value non-linearity, but does not clarify the reasons for this failure. For plain empirical reasons, the absolute-value has always been recommended as the best non-linearity for SF.

Here, we argue that one theoretical reason for the limited success of alternative implementations of SF is due to the fact that they cannot provide strong guarantees of preservation of data structure. If the absolute-value non-linearity is replaced with another non-linearity, such as sigmoid or ReLU, we likely lose the property of preservation of collinearity. Indeed, non-linearities such as sigmoid or ReLU do not induce in the original space representation filters with a regular conical shape, but they define wide arbitrary regions of space to be mapped onto a basis. Alternative non-linearities may preserve other structures, but these preservation properties must agree with the structure preserved by the other steps of SF, that is, steps A1, A3, A4. Thus, from a theoretical perspective, the absolute-value non-linearity is then an optimal choice for the SF algorithm, in that it preserves the property of collinearity which is also preserved by all the other steps of the algorithm, therefore guaranteeing the preservation of the overall structure defined by cosine neighbourhoodness. Alternative forms of the SF algorithm with different non-linearities will be reviewed more in detail in Section 3.5.1.

[2] https://www.kaggle.com/c/challenges-in-representation-learning-the-black-box-learning-challenge/forums/t/4717/1st-place-entry

3.3.7 Bounds on probability of preserving Euclidean neighbourhoodness

Interestingly, we can also define bounds on the probability of preserving Euclidean neighbourhoodness under very simplified assumptions. This bound depends mainly on the dimensionality of the original space $\mathbb{R}^M$ and on bounds on the region of space from which the samples $\mathbf{x}_i$ may be drawn.

Theorem 5. *Let $\mathbf{x}_1 \in \mathbb{R}^M$ be a point in the original space and let $R^{\mathbf{e}_1}_{\mathbf{x}_1}$ be a representation filter centred on $\mathbf{x}_1$, that is, $R^{\mathbf{e}_1}_{\mathbf{x}_1}(\mathbf{x}_1) = 0$. Let us now consider a point $\mathbf{x}_2 \in \mathbb{R}^M$ within the same representation cone, that is, a point such that $R^{\mathbf{e}_1}_{\mathbf{x}_1}(\mathbf{x}_2) < \epsilon$ for an arbitrarily small $\epsilon \in \mathbb{R}$, $\epsilon > 0$.*

Let us assume that: (i) points $\mathbf{x}_i$ distribute in a limited region of space bounded by a hyper-sphere of radius H; and, (ii) points $\mathbf{x}_i$ distribute uniformly in this limited region of space.

Then, given that $R^{\mathbf{e}_1}_{\mathbf{x}_1}(\mathbf{x}_2) < \epsilon$, it follows:

$$\frac{\sqrt{\pi}\cdot M\delta}{\left(\frac{H}{h}\right)^{M-1}}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)} \leq P\left(D_E\left[\mathbf{z}_1,\mathbf{z}_2\right]\leq\delta\right) \leq \frac{\sqrt{\pi}\cdot M\delta}{h}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)},$$

where $\delta \in \mathbb{R}$, $\delta > 0$ defines the neighbourhood of $\mathbf{x}_1$, h is the distance of $\mathbf{x}_1$ from the origin, and $\Gamma(\cdot)$ is the gamma function.

Proof. This proposition is proved geometrically, evaluating the limit of the ratio between the volume of a representation filter and the neighbourhood of the point $\mathbf{x}_1$.

Let us consider $\mathbf{x}_1 \in \mathbb{R}^M$ and let us define its neighbourhood as the set of points $\mathbf{x}_i$ within a hyper-sphere of radius δ, that is, $D_E\left[\mathbf{x}_1,\mathbf{x}_i\right] \leq \delta$. Let us consider now the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$ and let h be the distance of $\mathbf{x}_1$ from the origin. We first define the minimal representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$ as the hyper-cone of height h and radius δ inscribing the neighbourhood of $\mathbf{x}_1$. We also define a maximal representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$ as the hyper-cone of height H and, by trigonometry, radius $\Delta = H\cdot\frac{\delta}{h}$. For illustration, refer to the schema in Figure 3.3, where we represent this set-up in the case $M = 2$.

Let us now consider the point $\mathbf{x}_2$ sampled within the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$. Since the sampling probability is uniform within the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$, we can evaluate the probability of $\mathbf{x}_2$ to fall in the neighbourhood of $\mathbf{x}_1$ as the the volume of the neighbourhood of $\mathbf{x}_1$ normalized by the total volume of the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$.

Let us consider the neighbourhood of $\mathbf{x}_1$. Its volume can be computed as a function of the dimensions M and the radius δ:

$$V_{sphere}(M,\delta) = \mathcal{V}_M\delta^M,$$

where $\mathcal{V}_M$ is given by the following function:

$$\mathcal{V}_n = \frac{\pi^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}+1\right)}.$$

Let us now consider the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$. We bound this volume considering the

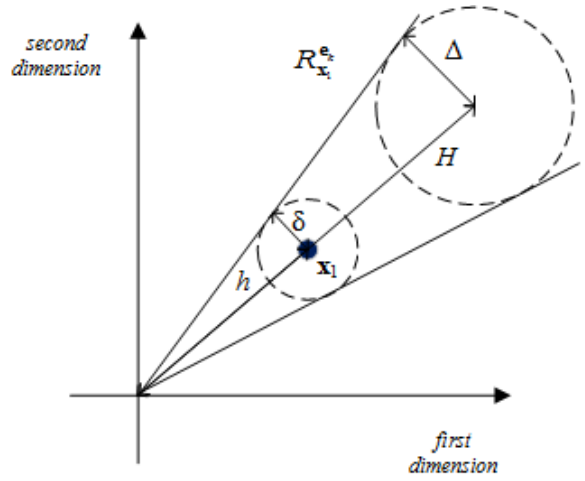

Figure 3.3: Schema of the the data point $\mathbf{x}_1$, the neighbourhood of $\mathbf{x}_1$ and the representation filter $R^{\mathbf{e}_k}_{\mathbf{x}_1}$ in two-dimensional space.

minimal and maximal hyper-cone described above. The volume of the hyper-cone depends on the volume of the lower-dimensional hyper-sphere in the base (Ball, 1997) and it can be computed and bounded as:

$$\begin{aligned} \frac{1}{M}\cdot h\cdot V_{sphere}(M-1,\delta) &\leq V_{cone}(M)\leq \frac{1}{M}\cdot H\cdot V_{sphere}(M-1,\Delta) \\ \frac{1}{M}\cdot h\cdot \mathcal{V}_{M-1}\cdot\delta^{M-1} &\leq V_{cone}(M)\leq \frac{1}{M}\cdot H\cdot \mathcal{V}_{M-1}\cdot\left(H\cdot\frac{\delta}{h}\right)^{M-1}. \end{aligned}$$

Let us now consider the ratio of the volume of the hyper-sphere and the volume of the hyper-cone:

$$\begin{aligned} \frac{\mathcal{V}_M\cdot\delta^M}{\frac{1}{M}\cdot H\cdot\mathcal{V}_{M-1}\cdot\left(H\cdot\frac{\delta}{h}\right)^{M-1}} &\leq \frac{V_{sphere}(M,\delta)}{V_{cone}(M)}\leq \frac{\mathcal{V}_M\cdot\delta^M}{\frac{1}{M}\cdot h\cdot\mathcal{V}_{M-1}\cdot\delta^{M-1}} \\ \frac{\sqrt{\pi}\cdot M\cdot\delta\cdot h^{M-1}}{H^M}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)} &\leq \frac{V_{sphere}(M,\delta)}{V_{cone}(M)}\leq \frac{\sqrt{\pi}\cdot M\cdot\delta}{h}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)}. \end{aligned}$$

Thus, given that $R^{\mathbf{e}_k}_{\mathbf{x}_1}(\mathbf{x}_2)\leq\epsilon$, it follows that

$$\frac{\sqrt{\pi}\cdot M\delta}{\left(\frac{H}{h}\right)^{M-1}}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)}\leq P\left(D_E\left[\mathbf{z}_1,\mathbf{z}_2\right]\leq\delta\right)\leq\frac{\sqrt{\pi}\cdot M\delta}{h}\cdot\frac{\Gamma\left(\frac{M+1}{2}\right)}{\Gamma\left(\frac{M+2}{2}\right)},$$

as stated. ■

Notice that this proof is based on two simplified assumptions. First, the region of the original space in which a point $\mathbf{x}_i$ can fall is limited; this assumption is reasonable because, practically, the range of any feature is always bounded, and, technically, features are often rescaled or normalized within bounded intervals. Secondly, a point $\mathbf{x}_i$ has a uniform probability of falling anywhere within the area defined by the representation filter $R^{\mathbf{e}_1}_{\mathbf{x}_1}$; this is clearly a simplified assumption because the pdf of the data $p(X)$ may have a very irregular distribution within

the area defined by the representation filter $R^{\mathbf{e}_1}_{\mathbf{x}_1}$; however, since such a pdf varies from case to case, assuming a uniform distribution, which is a distribution that maximizes the uncertainty, conforms to the principle of entropy maximization.

If these two assumptions are accepted, approximate bounds can be computed to evaluate the probability that SF will preserve relationships of Euclidean neighbourhoodness, together with cosine neighbourhoodness.

3.3.8 Sparse filtering for representation learning

Given the above results, we may now interpret SF as a soft clustering algorithm for representation learning.

Indeed, we may state that SF implicitly makes all the assumptions made by traditional soft clustering algorithms (see Section 2.2.4): (i) it is supposed to discover less noisy representations $\mathbf{z}_i$ whose pdf $p(Z)$ may automatically be closer to the true stochastic generating process with pdf $p(X^*)$; (ii) it expects the true pdf $p(X^*)$ to have a stronger correlation to the labels y_i; (iii) it models the true pdf $p(X^*)$ with a mixture model whose components are related to the bases $\mathbf{e}_i$; and, (iv) it relies on the cosine metric to evaluate relationships of neighbourhoodness in the original space $\mathbb{R}^M$. From this perspective, we can interpret the dimensionality of the learned space as the number of clusters for soft clustering, the bases as the cluster centroids in a space described by the cosine metric, the pursuit of the bases as the sequential process of updating the location of the centroids, and the learned representations $\mathbf{z}_i$ as the (stochastic) degree of membership of the original data samples $\mathbf{x}_i$ to each cluster.

Given this interpretation, we can align and meaningfully compare SF with other soft clustering algorithms for representation learning that use different metrics. The choice of an appropriate metric is critical for a distance-based clustering algorithm (Xing et al., 2003), and it expresses the understanding on which spatial directions encode relevant changes (Simard et al., 1998). It is natural then to compare SF with other standard algorithms which adopt the Euclidean metric to explain the data. Preserving the relationships of neighbourhoodness under the Euclidean metric means preserving the information conveyed by the pdf $p(X)$ in the representation space defined by the Cartesian product of the random variables $X_1, X_2, \ldots, X_M$. When the final goal is supervised learning, preserving this information makes sense if we expect that the structure of the data with respect to a set of labels $p(Y|X)$ is better explained by an Euclidean structure. In contrast, preserving the relationships of neighbourhoodness under the cosine metric means preserving information conveyed by the pdf $p(X)$ in the representation space defined by the projection into polar (or hyper-spherical) coordinates of the random variables $X_1, X_2, \ldots, X_M$. When the final goal is supervised learning, preserving such information makes sense if we expect that the structure of the data with respect to a set of labels $p(Y|X)$ is better explained by a radial structure.

3.3.9 Sparse filtering in information-theoretic terms

In conclusion, we can state that this analysis confirms our original thesis: SF satisfies both the informativeness principle and the infomax principle.

In particular, we showed that the informativeness principle is simply satisfied through the adoption of the proxy of sparsity, as shown in section 3.3.1.

The infomax principle, instead, is satisfied in a more subtle way, through the preservation of a precise structure underlying the data, that is, the radial structure of the data. Mutual information between the original representations $\mathbf{x}_i$ and the learned representations $\mathbf{z}_i$ is retained when the structure of the data is explained by the cosine neighbourhoodness, that is, in an ideal case, when all the information is carried by the angular coordinates of the data, as demonstrated in section 3.3.2.5. Indeed, the mutual information between the original and the learned representations can be formally expressed as: $MI[X;Z] = H_S[X] - H_S[X|Z]$. Given that the entropy of the distribution of the data $p(X)$ is fixed, the only way to maximize the mutual information is by minimizing the conditional entropy $H_S[X|Z]$. Since the learned representation $\mathbf{z}_i$ preserve all the information about the angular coordinates of the original representation $\mathbf{x}_i$, the uncertainty about $\mathbf{x}_i$ given $\mathbf{z}_i$ is minimized if the structure of the data has indeed a radial structure and information about the radial distance amounts to noise.

Interestingly, we can see the informativeness principle as the maximization of a purely syntactic measure of information; by minimizing the entropy of the learned representations, we minimize the purely formal notion of information in the data as the degree of unexpectedness of the data (see Section 2.2.1). On the other hand, we could see the infomax principle as a way to introduce a semantic constraint in an otherwise purely syntactic learning process; the definition of a specific structure to be preserved (in this case, a radial structure) means that we believe that such a structure is relevant and worthy to be preserved; the algorithm does not discover this structure, but it is designed, according to external human judgement, to retain it. This understanding crucially highlights the momentous role of the hidden assumption of cosine neighbourhoodness behind SF. The importance of uncovering this assumption, which we underlined in abstract terms in Section 2.2.2, is shown here in concrete terms: it is by discovering this assumption that we can understand the real dynamics of SF and the domain within which we could expect SF to work properly.

This summary concludes our theoretical analysis of SF. In the next section, we will test and validates these results experimentally.

3.4 Experimental Validation of Sparse Filtering

Based on the theoretical analysis provided in the previous section, we conduct a set of simulations aimed at verifying our theoretical results empirically. In order to make our results visualizable and easily understandable, we first conduct simple simulations on synthetic data in low dimensions; experiments on synthetic data in higher dimensions generalize our results but they do not add anything conceptually new to our conclusions. Finally, we further validate our theoretical findings with a number of benchmark data sets pertaining to real-world applications.

Section 3.4.1 starts by validating some of the basic properties of SF and of representation filters that we discussed in Section 3.3. Section 3.4.2 provides an experimental confirmation

of the central property of preservation of cosine neighbourhoodness. Section 3.4.3 extends the previous empirical results to higher-dimensional data sets. Section 3.4.4 illustrates the strengths and the limitations of SF in performing representation learning, and it contrasts the SF algorithm against the k-means algorithm (MacKay, 2003). Section 3.4.5 studies the use of SF with real-world data and uses our theoretical understanding to explain the success and the shortcomings of SF.

3.4.1 Properties of sparse filtering

First, we run simulations on elaborately designed toy data sets in order to validate our basic understanding of SF. These simulations aim at verifying: (i) the property of homo-representation of collinear points (see Section 3.3.2.3); (ii) the property of homo-representation of points with the same moduli (see Section 3.3.2.4); (iii) the usefulness of representation filters (see Section 3.3.4); and, (iv) the dynamics of pursuit of bases (see Section 3.3.3).

Data set. We generate a random set of data $\mathbf{X}$ of three samples ($N = 3$) in two-dimensional space ($M = 2$). Each point is generated using spherical coordinates: the radial distance ρ is sampled from a uniform distribution $\mathcal{U}(-5, 5)$; the angular coordinate θ is set to $\frac{\pi}{3}$ for the first two points and sampled from a uniform distribution $\mathcal{U}(0, \pi)$ for the third point.

Experimental protocol. A SF module is trained on $\mathbf{X}$ in order to learn a new representation of the data in two dimensions ($L = 2$). After training, a dense mesh of points $\mathbf{X}'$ in the original representation space $\mathbb{R}^M$ is created; each point $\mathbf{x}'$ is projected to its representation $\mathbf{z}'$ in the learned representation space $\mathbb{R}^L$, and the distance from each basis $\mathbf{e}_i$ in $\mathbb{R}^L$ is computed. The plot of each representation filter $R^{\mathbf{e}_i}$ is then shown as a two-dimensional contour plot in the original space $\mathbb{R}^M$.

Results. Figure 3.4 shows the state of SF before training. From the plots 3.4(b) and 3.4(d) we can immediately verify the property of homo-representation of collinear points; indeed, in the learned space $\mathbb{R}^L$ the collinear points occupy the same location and their matrix representation is the same. From the plots 3.4(e) and 3.4(f) we can verify the existence of representation filters in the original space $\mathbb{R}^M$ and appreciate several of the properties discussed above (existence of L representation filters; M-dimensionality of each representation filter; bounds of representation filters; and, complementarity of the representation filters). At the same time, the symmetric structure in the plots 3.4(e) and 3.4(f) validate the properties of homo-representation of points with the same moduli. Notice that, at this point, after the random initialization of the weight matrix $\mathbf{W}$, the quality of the representations generated by the untrained SF module is far from satisfactory.

Figure 3.5 shows the state of SF at the end of training. From the plots 3.5(b) and 3.5(d) we can see that the trained SF module has found an optimal solution that maps all the points onto bases; as expected, the collinear points are mapped to the same basis, while the third point is mapped onto the remaining basis. From the plots 3.5(e) and 3.5(f) we can verify our intuition about the pursuit of bases; indeed, training corresponded to a rotation of the representation

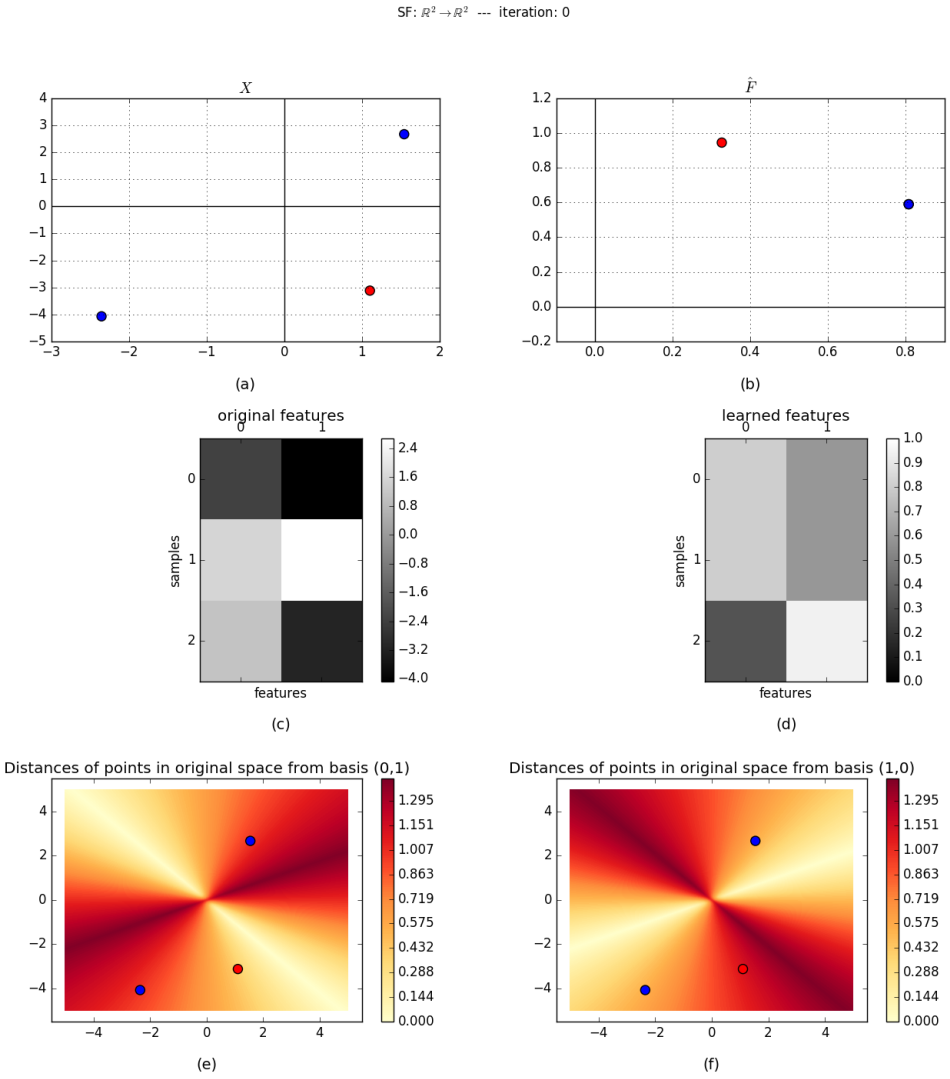

Figure 3.4: Experimental validation of the properties of SF (homo-representation of collinear points, homo-representation of points with the same moduli, representation filters).
This figure shows the state of SF at the beginning of the training. Data are generated as explained in the text (blue dots represent collinear points). (a) Data X in the original representation space $\mathbb{R}^M$ showing the original location of the samples; (b) data Z in the learned representation space $\mathbb{R}^L$ showing the location of the samples after processing through SF; (c) matrix plot of the original data $\mathbf{X}$ showing the value of the features of each of the three samples $\mathbf{x}_i$; (d) matrix plot of the learned representations $\mathbf{Z}$ showing the value of the features of each of the three samples $\mathbf{z}_i$; (e) contour plot of the first representation filter showing how far the learned representation of each point in the original space $\mathbb{R}^M$ is from the basis $\mathbf{e}_1 = [0, 1]^\top$ in the learned space $\mathbb{R}^L$; the white colour denotes points in $\mathbb{R}^M$ that are mapped close to the basis of $\mathbb{R}^L$, while the red colour denotes points in $\mathbb{R}^M$ that are mapped far from the basis of $\mathbb{R}^L$; (f) contour plot of the second representation filter showing how far the learned representation of each point in the original space $\mathbb{R}^M$ is from the basis $\mathbf{e}_2 = [1, 0]^\top$ in the learned space $\mathbb{R}^L$.
Notice that, before any learning, the representation Z generated by SF are not sparse and that the location of the representation filters appear to be random. However, the figure already illustrates the fact that collinear points (in blue) are mapped to identical representations and the fact that the SF algorithm instantiates conically-shaped representation filters.

filters in order to centre them on the available samples. Moreover, the same plots 3.5(e) and 3.5(f) also confirm the last properties of representation filters which we could not evaluate at the beginning of the simulation (association to the basis; closeness to a basis).

3.4.2 Preservation of cosine neighbourhoodness

Next, we run more simulations on other toy data sets in order to validate the properties of data structure preservation in SF. These simulations aim at verifying: (i) that SF preserves a structure defined by cosine neighbourhoodness (see Section 3.3.2.5); and, (ii) that the absolute-value non-linearity is crucial in preserving structure and substituting it with other non-linearities (such as sigmoid or ReLU) negates this property (see Section 3.3.6).

Data set. We generated a random set of data $\mathbf{X}$ of nine samples ($N = 9$) in two-dimensional space ($M = 2$). Each point is generated using spherical coordinates. All the points have a radial distance ρ sampled from a uniform distribution $\mathcal{U}(-5, 5)$. The first three points have an angular coordinate θ sampled from a uniform distribution $\mathcal{U}\left(\frac{\pi}{9} - \eta, \frac{\pi}{9} + \eta\right)$, the following three points have an angular coordinate θ sampled from a uniform distribution $\mathcal{U}\left(\frac{2\pi}{9} - \eta, \frac{2\pi}{9} + \eta\right)$, and the last three points have an angular coordinate θ sampled from a uniform distribution $\mathcal{U}\left(\frac{4\pi}{9} - \eta, \frac{4\pi}{9} + \eta\right)$. The parameter η is meant to represent a form of noise and its value is set to $\eta = \frac{\pi}{45}$. In this way, we generate three clusters of points, such that the cosine distances among the points belonging to the same cluster are small, while the cosine distances among points belonging to different clusters are large.

Experimental protocol. Three implementations of SF with different non-linearities (absolute-value, sigmoid, and ReLU[3]) are used to learn a new representation of the data in two dimensions ($L = 2$).

Results. Figure 3.6 shows the state of the modules of the three implementations of SF at the end of the training. From the plots 3.6(a)-3.6(c) we can immediately verify that SF with an absolute-value non-linearity preserves cosine neighbourhoodness. The plots of representation filters show that points with similar angular coordinates fall within the same representation filter. The matrix plot shows that points with similar angular coordinates are projected onto very similar representations; in other words, points that originally had a small cosine distance are projected onto almost identical representations. On the other hand, from plots 3.6(d)-3.6(i) we can easily see that SF with an alternative non-linearity does not preserve cosine neighbourhoodness. The optimization of the SF module with sigmoid and the ReLU non-linearity terminates very quickly without inducing representation cones, but defining instead large regions of the original space to be mapped onto a basis. Since these regions are not rigidly bounded (as in the case of the absolute-value non-linearity) several points are indistinctly mapped onto a basis. The matrix plots show that the representations computed by these

[3]ReLU has been implemented in a soft version, like the absolute-value: $\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \epsilon & \text{otherwise} \end{cases}$, where $\epsilon = 10^{-8}$

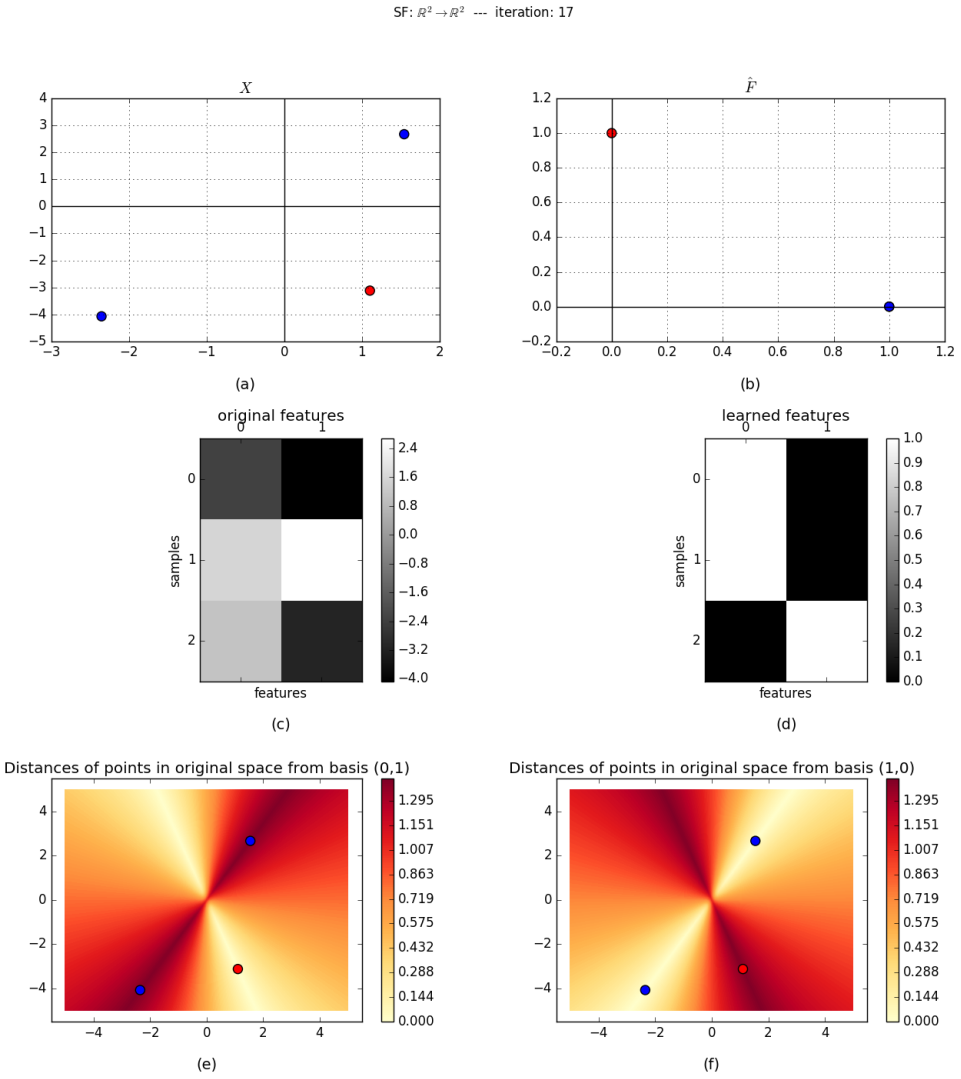

Figure 3.5: Experimental validation of the properties of SF (pursuit of bases).
This figure shows the state of SF at the end of the training. Data are generated as explained in the text. The meaning of the sub-plots is the same as in Figure 3.4.
Notice that, after learning, the representation Z generated by SF are now sparse and that the location of the representation filters indeed changed so that they are now centred on the data.

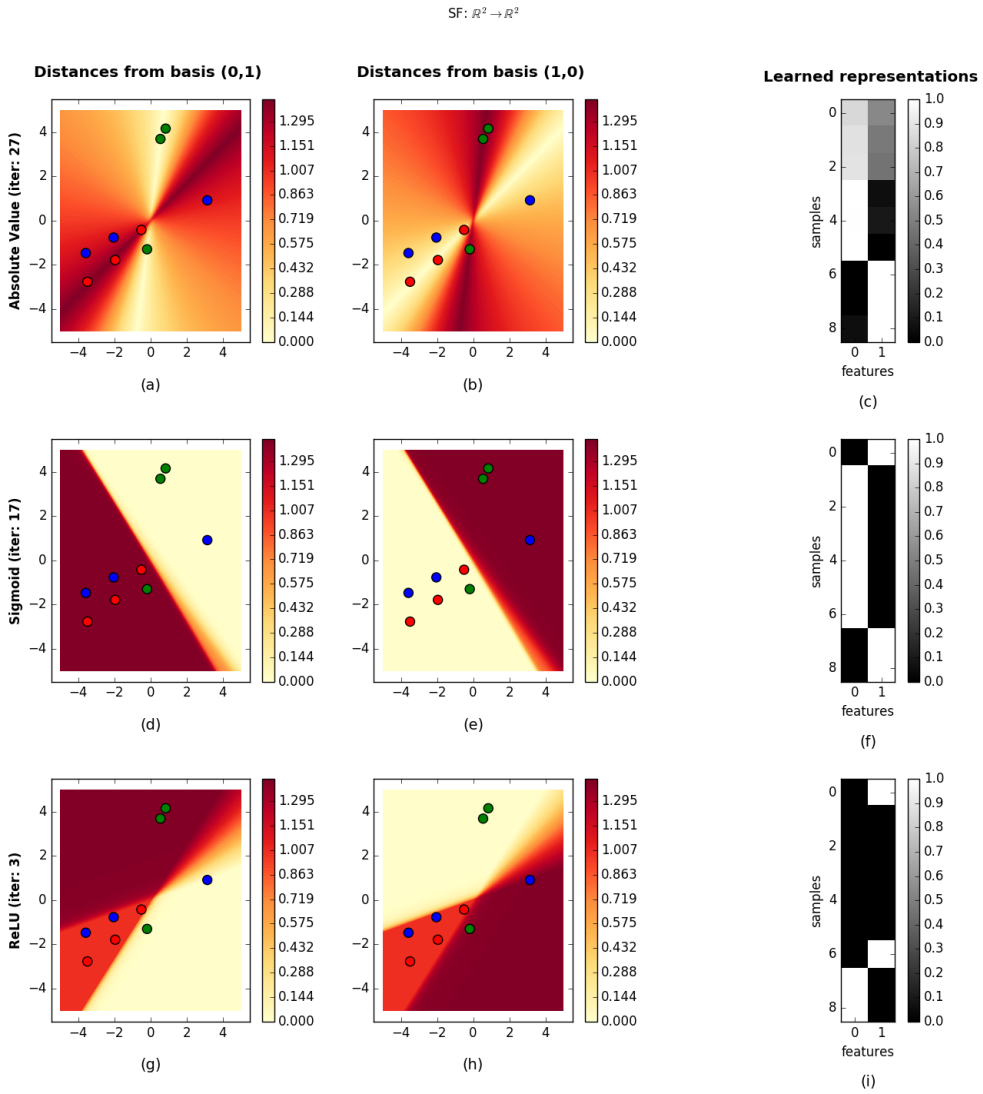

Figure 3.6: Experimental validation of the preservation of cosine neighbourhoodness.
Data are generated as explained in the text (first set of points in blue, second set of points in red, third set of points in green). (a, d, g) Plot of the first representation filter showing distances from the basis $\mathbf{e}_1 = [0, 1]^T$, respectively for the SF with absolute-value, sigmoid, and ReLU non-linearity; (b, e, h) plot of the second representation filter showing distances from the basis $\mathbf{e}_2 = [1, 0]^T$, respectively for the SF with absolute-value, sigmoid, and ReLU non-linearity; (c, f, i) matrix plot of the learned representations $\mathbf{Z}$, respectively for the SF with absolute-value, sigmoid, and ReLU non-linearity.
Notice that, where SF with an absolute-value non-linearity generates regular conically-shaped representation filters that preserve a radial structure, SF with sigmoid and ReLU non-linearity instantiate unstructured filters that can not preserve meaningful structures.

alternative SF modules are not related to the original cosine distances anymore; points originally belonging to the same cluster are mapped onto opposite representations, and, vice versa, points originally belonging to different clusters are mapped onto identical representations.

3.4.3 Preservation of cosine neighbourhoodness in high-dimensions

Furthermore, we run an additional set of simulations on higher-dimensional synthetic data sets in order to confirm that the conclusions we drew from the previous experiments are not limited to the two-dimensional case. These simulations aim at verifying: (i) that, even in high-dimensions, SF preserves a structure defined by cosine neighbourhoodness (as we discussed in Section 3.4.2); and, (ii) that, even in high dimensions, representation filters explain the dynamics of SF (as we observed in Section 3.4.1).

Data set. To verify the preservation of cosine neighbourhoodness, we generated a first random data set $\mathbf{X}'$ of twelve samples ($N = 12$) in four-dimensional space ($M = 4$). The twelve samples are divided into four sets of three collinear points. Each point is generated using spherical coordinates. Each set is generated from a cluster having a different angular coordinate ϕ_i, $1 \leq i \leq 4$; all the clusters are described by a Gaussian distribution $\mathcal{N}(\mu_i, K)$ with a specific mean vector μ_i and a diagonal covariance matrix K whose elements on the main diagonal are 0.05. The first set of points is centred at $\mu_1 = [1, 1, 2, 2]^\top$, the second at $\mu_2 = [-1, 3, 4, 4]^\top$, the third at $\mu_3 = [2, 2, 4, 2]^\top$ and the fourth at $\mu_4 = [-2, 4, 1, 1]^\top$.

To validate the generation of representation filters in higher dimensions, we generate a second data set $\mathbf{X}''$ of ten samples ($N = 10$) in two-dimensional space ($M = 2$). Reducing the dimensionality of the original space is necessary in order to visualize the representation filters, which would be hard or impossible to represent in higher dimensions. All the ten points are generated simply by randomly sampling their coordinates from a uniform distribution $\mathcal{U}(-5, 5)$.

Experimental protocol. Both data sets are processed using SF in order to learn new representations of the data in eight-dimensional space ($L = 8$).

Results. Figure 3.7 shows the matrix plot of the original representations $\mathbf{X}'$ in the original space $\mathbb{R}^4$ and of the learned representations $\mathbf{Z}'$ in the learned space $\mathbb{R}^8$. While in the original representations the clustering of the samples in four sets defined by their radial coordinates is hard to detect, in the learned representations this structure is immediately apparent. Indeed all the samples belonging to the same clusters are mapped onto the same representation; in particular, samples from the first and the fourth clusters are mapped onto perfectly 1-sparse representations, that is onto the $\mathbf{e}_1$ and the $\mathbf{e}_5$ bases; samples from the second and the third clusters are mapped onto sub-optimal representations, respectively a 4-sparse and a 2-sparse representation. In general, even if the training procedure may take longer to converge to a solution, the overall dynamics are the same. Figure 3.8 shows the representation filters generated while learning a projection of the original representations $\mathbf{X}''$ from the original space $\mathbb{R}^2$ onto the learned representations $\mathbf{Z}''$ in the learned space $\mathbb{R}^8$. Even if the conical shape of the representation filters is less accentuated due to the difficulty of mapping an unstructured

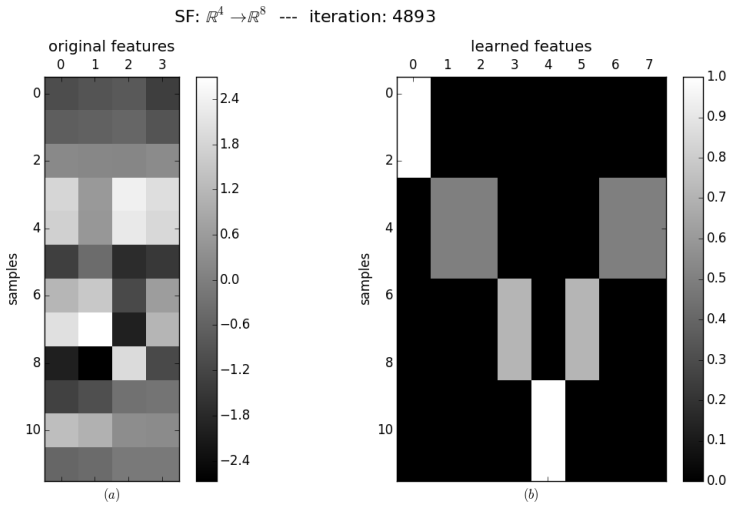

Figure 3.7: Experimental validation of the preservation of cosine neighbourhoodness in high dimensions.
Data are generated as explained in the text. (a) Matrix plot of the original representations $\mathbf{x}_i$; (b) Matrix plot of the learned representations $\mathbf{z}_i$.

data set onto the bases of $\mathbb{R}^8$, the overall dynamics of SF are still easily detectable: individual representation filters move across the original space in order to be centred on specific data samples and project such data samples onto bases of the learned space.

3.4.4 Sparse filtering for representation learning

In the following set of simulations, we compare SF against another unsupervised algorithm, the soft k-means algorithm (MacKay, 2003) in order to show under which conditions SF is a good choice for processing data. These simulations aim at verifying the following intuitive implication: if the structure of the data with respect to a specific set of labels $p(Y|X)$ is better explained by the cosine metric, then SF is likely to be a good option for unsupervised learning.

In our comparison, we measure SF against the soft k-means algorithm. We chose this algorithm for the following reason: (i) like SF, the soft k-means algorithm is a soft clustering algorithm producing sparse representations; (ii) the algorithm is based on the Euclidean metric, thus providing a different interpretation of the data from SF; and, (iii) k-means is a well-known and easy-to-interpret algorithm (even if analogous results may be obtained with other algorithms, such as GMM).

Data set. To validate our hypothesis, we generate two data sets, $\mathbf{X}^{\mathbf{Euclid}}$ and $\mathbf{X}^{\mathbf{cosine}}$. The data set $\mathbf{X}^{\mathbf{Euclid}}$ contains data where $p(Y|X)$ is explained by the Euclidean metric. It is composed of nine samples ($N = 9$) in two dimensions ($M = 2$). The first three points are sampled from a multivariate normal distribution $\mathcal{N}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} .05 & 0 \\ 0 & .05 \end{bmatrix}\right)$; the second three points are

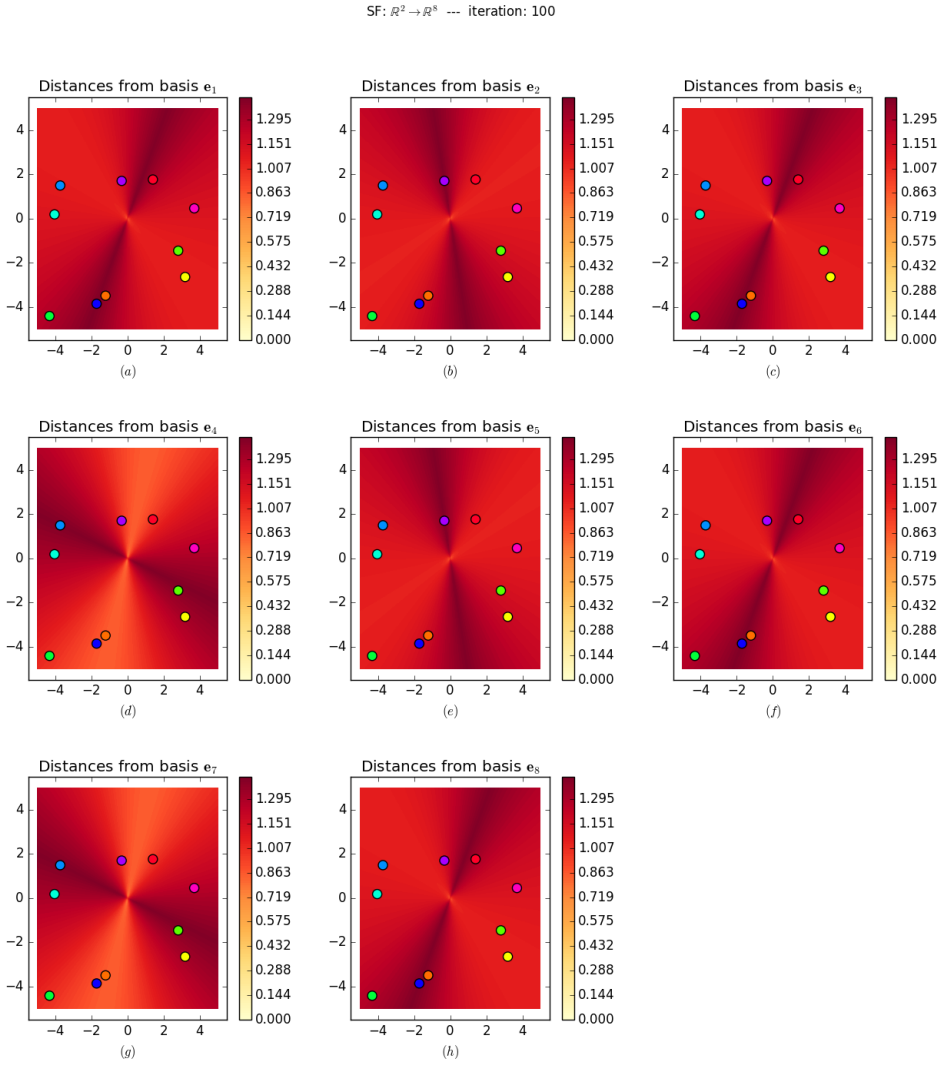

Figure 3.8: Representation filters at the end of learning in high dimensions.
Data are generated as explained in the text. Colours do not have any meaning, as a random colour was assigned to each point just for distinguishing them. (a-h) Representation filters associated with the bases $\mathbf{e}_k$.

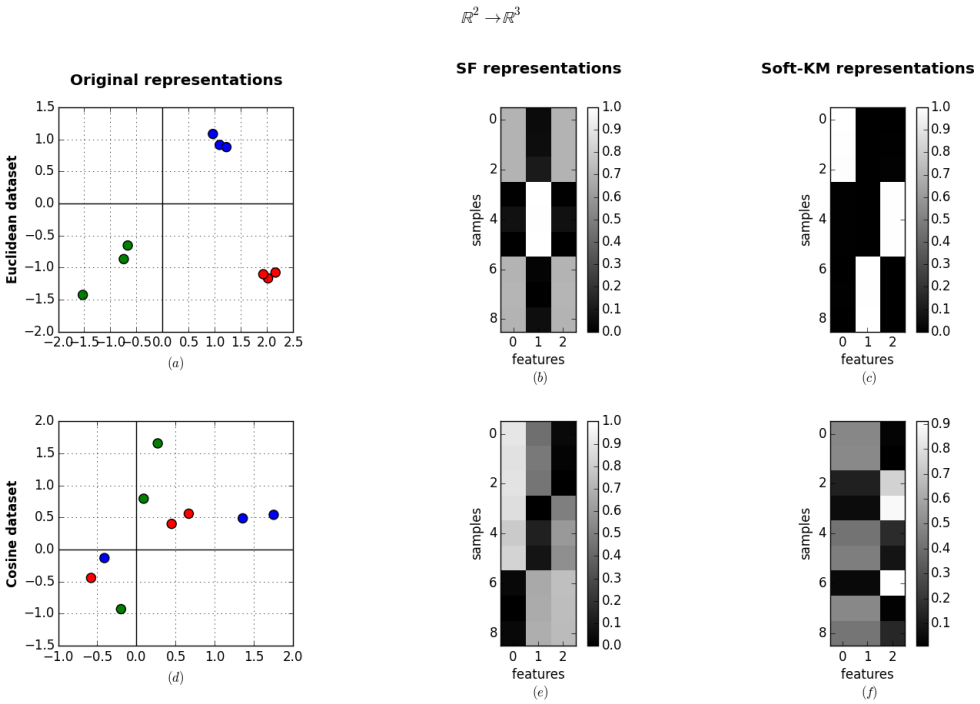

Figure 3.9: Representation learning using SF and k-means on data with Euclidean and cosine data structure.
Data are generated as explained in the text (first set of points in blue, second set of points in red, third set of points in green). (a, d) Samples in the original space; (b, e) matrix plot of the representations learned by SF; (c, f) matrix plot of the representations learned by soft k-means.

sampled from a multivariate normal distribution $\mathcal{N}\left(\begin{bmatrix} 2 \\ -1 \end{bmatrix}, \begin{bmatrix} .05 & 0 \\ 0 & .05 \end{bmatrix}\right)$; the last three points are sampled from a multivariate normal distribution $\mathcal{N}\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} .05 & 0 \\ 0 & .05 \end{bmatrix}\right)$. The data set $\mathbf{X}^{\mathbf{cosine}}$ contains data where $p(Y|X)$ is explained by the cosine metric. The data is generated following the same protocol used in the simulation in Section 3.4.2.

Experimental protocol. SF is used to learn a new representation of the data in three dimensions ($L = 3$). The soft k-means algorithm is equivalently instantiated using three centroids.

Results. From Figure 3.9, we can see that our understanding of SF is correct: if $p(Y|X)$ is better explained by the cosine metric, then SF produces a good representation; otherwise, if $p(Y|X)$ is better explained by the Euclidean metric, then it is reasonable to opt for a different unsupervised learning algorithm, such as soft k-means. In the case of the data set with Euclidean structure, plot 3.9(b) shows that SF is not able to preserve the identity of the generating clusters, and indeed it maps samples from the first and the third clusters onto the same representation

(because of their collinearity); instead, plot 3.9(c) shows that soft the k-means algorithm maps points from different clusters onto different representations. In contrast, in the case of the data set with cosine structure, plot 3.9(e) shows that SF preserves the identity of the generating clusters, while plot 3.9(f) shows that the soft k-means algorithm is unable to map samples from the same cluster onto consistent representations.

3.4.5 Sparse filtering on real data sets

In this last set of simulations we apply our discoveries about SF to real-world data sets to further verify our results. Once again, these experiments aim at validating the connection between the radial structure of the data and the success of SF. In the first simulation, we extend the result that we proved in Section 3.4.4 for toy data sets to real data sets; that is, we verify the direct implication: *if* the structure of the data with respect to a specific set of labels $p(Y|X)$ is better explained by the cosine metric, *then* SF is likely to be a good option for unsupervised learning. In the second simulation, we validate, instead, the reverse implication: *if* SF happens to be a good option for unsupervised learning, *then* the structure of the data with respect to a specific set of labels $p(Y|X)$ is likely to be better explained by the cosine metric.

Notice that when dealing with real data sets, it is very challenging to assess the structure of the data. In low dimensions, with few samples and with the simplified assumption that all the data belonging to a given class are generated by a single highly localized cluster (as in the previous simulations), a simple visualization of the data is enough to understand which metric is underlying the data. Thanks to these simplified settings, a straight computation of distances among samples belonging to the same class is sufficient to decide which metric best describes the data. However, when considering real data sets, we have to deal with samples in high dimensions, with a large number of samples and with the fact that samples belonging to the same class may be generated by different clusters spread throughout the space; in this case, it is not possible to rely on visualization any more. Being unable to produce any useful visualization, we can not assess precisely the structure of the data and provide a precise and thorough interpretation of the results, as we did in the case of synthetic data. Instead, we have to rely on indirect synthetic measures, such as classification accuracy, to get insights on the structure of the data. Specifically, in order to explore high-dimensional data, we decided to rely on the KNN algorithm. We implemented two versions of KNN, one selecting k neighbours according to the Euclidean distance and one selecting k neighbours according to the cosine distance[4]. If $p(Y|X)$ is better explained by the Euclidean distance, we expect KNN with the Euclidean metric to provide better results; alternatively, if $p(Y|X)$ is better explained by the cosine distance, we expect KNN with the cosine metric to provide better results.

[4]The KNN using cosine distance has been implemented relying on the "trick" that the cosine distance between vectors $\mathbf{u}, \mathbf{v}$ is the same as the Euclidean metric on the ℓ_2-normalized vectors. Therefore, we perform an ℓ_2-normalization of each data sample and then we run KNN with Euclidean distance, re-using off-the-shelf KNN code optimized for the Euclidean metric.

3.4.5.1 First simulation

Data set. The Berlin Emotional (EMODB) data set is a well-known audio data set in the emotional speech recognition (ESR) community (Burkhardt et al., 2005); it contains recordings of ten German actors expressing seven different types of emotions (see Appendix B for a more detailed description of the data set). We opted for this emotional speech data set to validate the direct implication between data structure and effectiveness of SF for the following reasons. (i) Samples in EMODB naturally lend themselves to alternative labellings; in particular, the same data may be used both for speaker recognition (using subject labels) and for emotion recognition (using emotional content labels). (ii) The same set of Mel-frequency cepstrum (Childers et al., 1977) coefficient (MFCC) features may reasonably be used both for speaker recognition and for emotion recognition; indeed, MFCC features were primarily designed for speaker recognition, but they proved to be relevant for emotion recognition as well (Wu et al., 2010; Schuller et al., 2011). Using the same features we can then explore emotional speech data under different labelling.

Experimental protocol. We first explore the structure of the data with respect to the two different labelling systems in order to evaluate whether the Euclidean distance or the cosine distance better explains the structure of the data. The KNN algorithm is run with different values of neighbours ($k = \{2, 3, 5, 7, 10, 15, 20, 25, 50, 75, 100\}$); for each configuration of KNN fifty simulations are executed; in each simulation the data set is randomly partitioned into a training data set (900 samples) and a test data set (311 samples); KNN is then trained and tested using the two available metrics.

After this analysis, we use both an Euclidean-based unsupervised learning algorithm, Gaussian mixture model (Bishop, 2007), and a cosine-based unsupervised learning algorithm, SF, to project the data into an L-dimensional space. We opted for the GMM algorithm because it is based on the Euclidean metric and yields better performance than the soft k-means algorithm. After processing the data, we run a simple linear SVM classifier on the processed data and we analyse how our observations on the structure of the data relate with the actual classification performance. We consider several values of dimensionality ($L = \{2, 3, ..., 40\}$); for each configuration, fifty simulations are executed; as before, in each simulation the data set is randomly partitioned into a training data set (900 samples) and in a test data set (311 samples).

Results. Figure 3.10(a) shows that the structure of EMODB data with respect to emotional labels is better explained by the Euclidean distance. This result is further confirmed by the classification with the linear SVM module in Figure 3.10(b). Even if the contribution of preprocessing via GMM or SF does not improve the results in absolute terms, we can still observe that classification using the GMM-processed data with low learned dimensionality ($L \leq 15$) returns an accuracy that is significantly better than using SF-processed data (Wilcoxon signed-rank test, p-value $p = 5 \cdot 10^{-85}$); in higher dimensions, instead, the classification with SF-processed data approaches and overtakes the accuracy obtained using GMM-processed data. In general, in low dimensions, the Euclidean structure assumed by GMM explains the data better; in high dimensions, SF provides good results (most likely thanks to the property of sparsity)

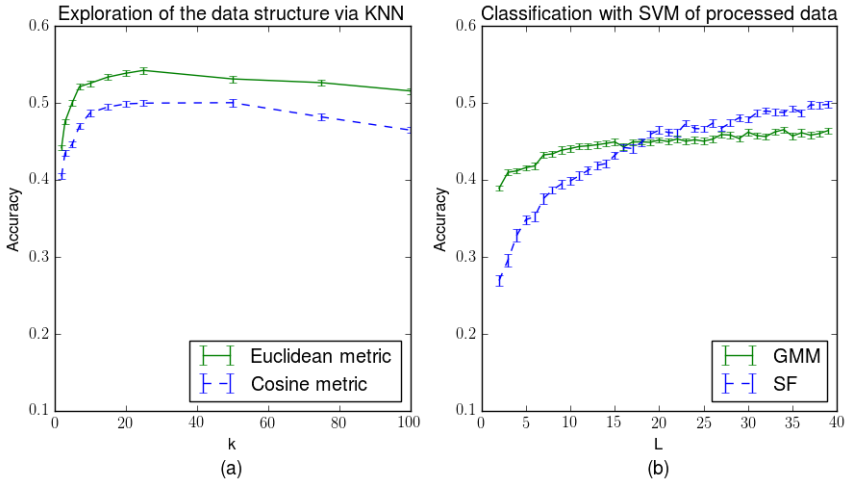

Figure 3.10: Analysis of the data structure and the classification of the EMODB data set with respect to emotion labels.
Classification is performed as explained in the text. (a) Exploration of the data via KNN with Euclidean metric (green line) and with cosine metric (blue line); (b) Classification using a linear SVM after processing with a GMM algorithm (green line) and with SF (blue line). The plot shows the average accuracy and the standard error of SVM (over fifty simulations).

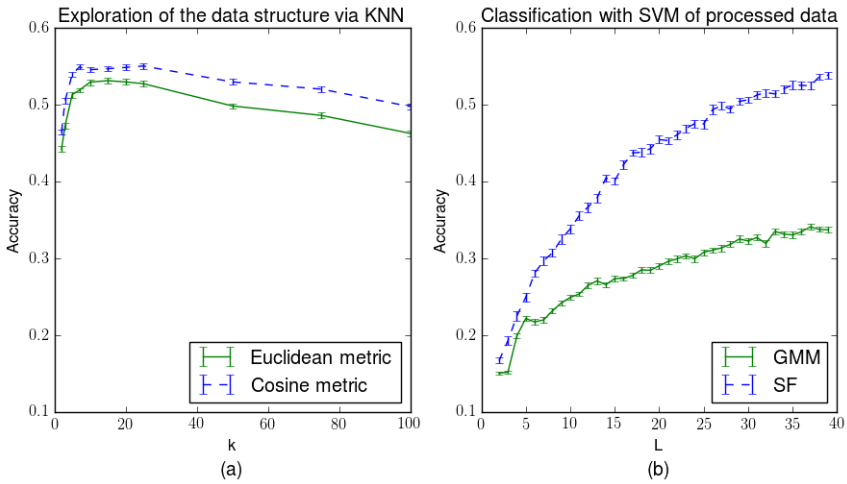

Figure 3.11: Analysis of the data structure and the classification of the EMODB data set with respect to subject labels.
The meaning of the sub-plots is the same as in Figure 3.10.

but the gap between the accuracy provided by the two representations remains limited. On the other hand, Figure 3.11(a) shows that the structure of EMODB data with respect to the speaker identity labels is better explained by the cosine distance. This result is further confirmed by the classification with the linear SVM module in Figure 3.11(b). Classification using the SF-processed data returns, for all learned dimensionality, an accuracy that is significantly better than GMM-processed data (Wilcoxon signed-rank test, p-value $p = 4 \cdot 10^{-307}$). The assumption of the cosine metric allows SF to explain the data much better, as is evident from the large gap between the accuracy provided by the two representations.

These results confirm a connection between the radial structure of the data with respect to a set of labels and the usefulness of SF.

3.4.5.2 Second simulation

Data set. The Kaggle Black Box Learning Challenge (KBBLC) data set is a visual data set made up of obfuscated images of house numbers; the original images are taken from the well-known Street View House Numbers (SVHN) data set (Netzer et al., 2011). Each sample in the KBBLC data set contains a single obfuscated digit and it is accompanied by a label specifying the value of the digit. We opted to validate the reverse implication between data structure and effectiveness of SF on this data set for the following reasons. (i) SF provided state-of-the-art performance in the competitive KBBLC contest organized during the 2013 International Conference on Machine Learning, thus showing that SF was a particularly suitable choice for this data set. (ii) The KBBLC data set is available with labels. During the challenge the authors provided obfuscated data without labels; however, after the challenge they revealed the original source of the data[5] and they released the code they had used for obfuscation[6]. Thanks to this information, we were able to retrieve a large amount of data and obfuscate it, and thus recreate the original conditions of the challenge. However, differently from the challenge, we retain the labels in order to explore the structure of the data. (iii) During the challenge, the original samples from the data sets were processed without undergoing operations of windowing or convolution. Since SF was directly applied to the samples, we can analyse the structure of the samples straightforwardly. This condition is not always true. If we consider other image data sets on which SF provided good results, such as CIFAR-10 (Krizhevsky and Hinton, 2009) or STL-10 (Coates et al., 2011), SF was not applied to the original samples but to random patches extracted from the images; in this case, we should not analyse the data structure of the original samples, but the data structure of the patches. However, patches are not labelled, which hinders the ability to carry out an analysis of the data structure.

Experimental protocol. In exploring the structure of the data (with respect to the digit labels), we aim at evaluating whether the Euclidean distance or the cosine distance better explains $p(Y|X)$. We run the KNN with the same settings as in the previous experiment. In each simulation a random subset of 10000 samples from the data set was selected and then

[5] http://ufldl.stanford.edu/housenumbers/

[6] https://www.kaggle.com/c/challenges-in-representation-learning-the-black-box-learning-challenge/forums/t/5167/the-data

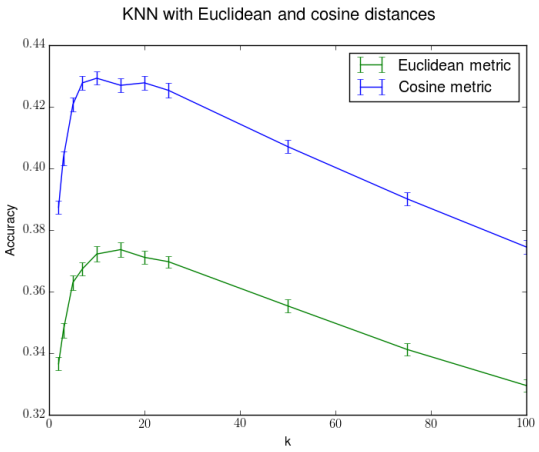

Figure 3.12: Analysis of the data structure of the Kaggle Black Box Learning Challenge data set.
The KNN with Euclidean metric (green line) and with cosine metric (blue line) has been used to explore the structure of the data. The plot shows the average accuracy and the standard error of KNN (over five simulations).

partitioned into a training data set (9000 samples) and a test data set (1000 samples). KNN was then trained and tested using one of the two available metrics.

Results. Figure 3.12 confirms our intuition. For all the different values of k we considered, the cosine distance proved to be a better metric to explain the structure of the data in the Kaggle Black Box Learning Challenge. This provides an explanation why SF proved so useful with the KBBLC data, when compared to other standard unsupervised learning algorithms, especially those based on the Euclidean metric. This result agrees with the fact that the Euclidean metric is not a suitable metric for measuring distances among samples of digits represented in the pixel space; other distances less sensitive to irrelevant transformations, such as tangent distance (Simard et al., 1998), are known to be better choices.

The experiments carried out in this section agreed with our theoretical results and they clearly confirmed the properties, the strengths and the weaknesses of the standard SF algorithm. In the next section we will explore the possibility of defining alternative forms of the SF algorithm.

3.5 Alternative Feature Distribution Learning Algorithms

In this section we now consider alternative forms of FDL algorithms, analysing both existing and novel algorithms. The conceptual and theoretical framework that we developed to study and understand SF will be useful to evaluate other algorithms as well, and to highlight their strengths and weaknesses.

Section 3.5.1 first considers potential new algorithms that we may develop starting from the original SF algorithm. Section 3.5.2 takes into consideration an already-existing class of

algorithms and reviews it under the lens of the FDL framework.

3.5.1 Sparse filtering-like algorithms

In Section 3.3 we introduced a set of criteria and tools to study the SF algorithm: we proposed to evaluate its dynamics in relation to the principles of informativeness (Section 3.3.1) and infomax (Section 3.3.2); we explained the guarantees of the algorithm in terms of preservation of the data structure (Section 3.3.2.5); and we introduced the tool of representation filters to inspect its inner workings (Section 3.3.4). We now rely on these criteria and tools to analyse the possibility of defining new SF-like algorithms.

3.5.1.1 Alternative forms of sparse filtering

The simplest way to define new SF-like algorithms is to act on the original algorithm and modify it. In order to decide what can be changed and edited, it is important to recall that new versions of the algorithm will still have to satisfy both the informativeness and the the infomax principle.

We showed that the informativeness principle is satisfied through the proxy of sparsity. Alternative forms of SF must then retain the ability to learn sparse representations. Now, as discussed in Section 3.3, the two ℓ_2-normalization steps (A3, A4) are integral for the learning of sparsity. They project the original data points onto a unit hypersphere and they allow for sparsification through the optimization carried out in step A5. Modifying these steps would then affect the intrinsic design principles of SF, leading to an essentially different FDL algorithm. For the sake of learning sparse representations and satisfying the informativeness principle, we keep steps A3 and A4 unmodified in the following study.

Once the idea of editing the normalization steps of SF is excluded, room for change becomes restricted mainly to step A2, that is the choice of non-linearity. In Section 3.3 we have highlighted how the choice of non-linearity is strictly connected to the type of structure preserved by SF: in Section 3.3.6 we hinted at the fact that SF implementations using alternative non-linearities may not preserve cosine neighbourhoodness, and in Section 3.4.2 we showed this to be the case. In general, the shape of the representation filters generated by SF is particularly sensitive to the choice of the non-linearity, and the conical or hyper-conical shape we analysed was due to the specific choice of the absolute-value non-linearity. Different non-linearities would induce different representation filters which, in turn, could preserve different data structures.

Virtually any non-linear function could be plugged in step A2, as long as the function is derivable. Derivability is required in order to perform back-propagation during training. Alternative non-linearities may then allow us to satisfy the infomax principle in different ways.

Notice that changing the non-linearity in SF does not change the intrinsic representational power of SF, but only changes the type of filters learned by SF. In other words, it does not change the representation learning dynamics of SF, but simply modifies the type of filters learned.

The preliminary empirical results in Section 3.4.2 seemed to suggest that standard non-linearity from the neural networks literature do not fare very well when used in the SF algorithm.

We will now provide formal arguments that alternative implementations of SF using sigmoid or ReLU non-linearities are both unable to preserve cosine or Euclidean distances.

3.5.1.2 Rectified-linear-unit sparse filtering

ReLU non-linearities have become very popular in neural network literature in recent years due to their simplicity and computational efficiency. Absolute-value and ReLU non-linearities share a superficial similarity, as their behaviour on positive values is the same. It may be tempting, then, to swap the absolute-value non-linearity for a ReLU non-linearity.

Let us define the *ReLU SF* algorithm through the following transformation:

$$\mathbf{Z} = \ell_{2,col}\left(\ell_{2,row}\left(\text{ReLU}\left(\mathbf{WX}\right)\right)\right),$$

where the ReLU non-linearity is implemented as soft ReLU: $softReLU(x) = \begin{cases} x & \text{if } x > 0 \\ \epsilon & \text{otherwise} \end{cases}$, for a small $\epsilon = 10^{-8}$. Now, it is immediate to prove the following proposition.

Proposition 3. *Let us consider the ReLU SF algorithm and let $\mathbf{x}_i$ be points in the original space $\mathbb{R}^M$. Then, the transformations $f_{A1:A4}$, where A2 is now the ReLU non-linearity, do not preserve the structure of the data described either by the Euclidean metric or by the cosine metric.*

Proof. We divide this proposition in two parts and we prove each one by counterexample.

Let us focus first on the non-preservation of the Euclidean metric. Let us consider the case in which $\mathbf{x}_1$ is a vector such that $x_{1,j} = -\frac{3}{\sqrt{2}}$, $\forall j$, $1 \leq j \leq M$, $\mathbf{x}_2$ is another vector such that $x_{2,j} = -\frac{1}{\sqrt{2}}$, $\forall j$, $1 \leq j \leq M$, $L = M$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is:

$$D_E\left[\mathbf{x}_1, \mathbf{x}_2\right] = \sqrt{\sum_{j=1}^{M}\left(-\frac{3}{\sqrt{2}} + \frac{1}{\sqrt{2}}\right)^2} = \sqrt{2M}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{aligned}
f_{A1}\left(\mathbf{x}_1\right) &= \mathbf{I}\mathbf{x}_1 = \mathbf{x}_1 & f_{A1}\left(\mathbf{x}_2\right) &= \mathbf{I}\mathbf{x}_2 = \mathbf{x}_2 \\
f_{A2}\left(\mathbf{x}_1\right) &= \text{ReLU}\left(\mathbf{x}_1\right) = [\epsilon] & f_{A2}\left(\mathbf{x}_2\right) &= \text{ReLU}\left(\mathbf{x}_2\right) = [\epsilon] \\
f_{A3}\left([\epsilon]\right) &= \left[\frac{\epsilon}{\sqrt{N}\epsilon}\right] = \left[\frac{1}{\sqrt{N}}\right] & f_{A3}\left([\epsilon]\right) &= \left[\frac{\epsilon}{\sqrt{N}\epsilon}\right] = \left[\frac{1}{\sqrt{N}}\right] \\
f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right) &= \left[\frac{\sqrt{N}}{\sqrt{N}\sqrt{L}}\right] = \left[\frac{1}{\sqrt{L}}\right] = \mathbf{z}_1 & f_{A4}\left(\left[\frac{1}{\sqrt{N}}\right]\right) &= \left[\frac{\sqrt{N}}{\sqrt{N}\sqrt{L}}\right] = \left[\frac{1}{\sqrt{L}}\right] = \mathbf{z}_1.
\end{aligned}$$

Thus, $f_{A1:A4}\left(\mathbf{x}_1\right) = \mathbf{z}_1$ and $f_{A1:A4}\left(\mathbf{x}_2\right) = \mathbf{z}_1$. Now, the Euclidean distance between the vectors $f_{A1:A4}\left(\mathbf{x}_1\right)$ and $f_{A1:A4}\left(\mathbf{x}_2\right)$ is:

$$D_E\left[\mathbf{z}_1, \mathbf{z}_1\right] = 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. This proves the first part of the proposition.

Let us focus now on the non-preservation of the cosine metric. Let us consider the case in which $\mathbf{x}_1$ is a vector such that $x_{1,j} = \frac{1}{2^j}$, $\forall j$, $1 \leq j \leq L$, $\mathbf{x}_2$ is another vector such that $\mathbf{x}_2 = -\mathbf{x}_1$, $L = M = 2$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The cosine distance between the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is:

$$D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] = 1 - \left| \frac{\sum_{j=1}^{M} x_{1,j} x_{2,j}}{\sqrt{\sum_{j=1}^{M} x_{1,j}^2} \sqrt{\sum_{j=1}^{M} x_{2,j}^2}} \right| = 0.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{array}{cc} f_{A1}\left(\mathbf{x}_1\right) = \mathbf{I}\mathbf{x}_1 = \mathbf{x}_1 & f_{A1}\left(\mathbf{x}_2\right) = \mathbf{I}\mathbf{x}_2 = \mathbf{x}_2 \\ f_{A2}\left(\mathbf{x}_1\right) = \text{ReLU}\left(\mathbf{x}_1\right) = \mathbf{x}_1 & f_{A2}\left(\mathbf{x}_2\right) = \text{ReLU}\left(\mathbf{x}_2\right) = [\epsilon] \\ f_{A3}\left(\mathbf{x}_1\right) = \left[\frac{x_{1,j} 2j}{\sqrt{1+2^{2j}\epsilon^2}}\right] = \left[\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}\right] & f_{A3}\left([\epsilon]\right) = \left[\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}\right] \\ f_{A4}\left(\left[\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}\right]\right) = \left[\frac{\frac{1}{\sqrt{1+2^{2j}\epsilon^2}}}{\sqrt{\sum_{j=1}^{L} \frac{1}{1+2^{2j}\epsilon^2}}}\right] = \mathbf{z}_1 & f_{A4}\left(\left[\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}\right]\right) = \left[\frac{\frac{\epsilon 2j}{\sqrt{1+2^{2j}\epsilon^2}}}{\sqrt{\sum_{j=1}^{L} \frac{2^{2j}\epsilon^2}{1+2^{2j}\epsilon^2}}}\right] = \mathbf{z}_2. \end{array}$$

Thus, $f_{A1:A4}\left(\mathbf{x}_1\right) = \mathbf{z}_1$ and $f_{A1:A4}\left(\mathbf{x}_2\right) = \mathbf{z}_2$, where $\mathbf{z}_1 = \left[\begin{array}{cc} \frac{\sqrt{1+16\epsilon^2}}{\sqrt{2+20\epsilon^2}} & \frac{\sqrt{1+4\epsilon^2}}{\sqrt{2+20\epsilon^2}} \end{array}\right]$ and $\mathbf{z}_2 = \left[\begin{array}{cc} \frac{\sqrt{1+16\epsilon^2}}{\sqrt{5+32\epsilon^2}} & \frac{2\sqrt{1+4\epsilon^2}}{\sqrt{5+32\epsilon^2}} \end{array}\right]$. Now, the cosine distance between the vectors $f_{A1:A4}\left(\mathbf{x}_1\right)$ and $f_{A1:A4}\left(\mathbf{x}_2\right)$ is:

$$D_C\left[\mathbf{z}_1, \mathbf{z}_2\right] \neq 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the cosine metric. This proves the second part of the proposition. ■

Despite the apparent similarity, there is a critical difference between the absolute-value non-linearity and the ReLU non-linearity, as the latter cannot preserve either cosine distance or collinearity (as entailed by Proposition 3). Differently from standard SF, ReLU SF is then unable to preserve the structure of the data defined by cosine neighbourhoodness. This was confirmed by the experiment in Section 3.4.2, where we observed that the ReLU non-linearity partitions the original space with linear sharp boundaries.

3.5.1.3 Sigmoid sparse filtering

Another traditional non-linearity from the neural network literature with a very wide application is the sigmoid non-linearity. Given its success in other applications we may consider once again the possibility of swapping the absolute-value non-linearity for a sigmoid non-linearity.

Let us define the *sigmoid SF* algorithm through the following transformation:

$$\mathbf{Z} = \ell_{2,col}\left(\ell_{2,row}\left(\sigma\left(\mathbf{W}\mathbf{X}\right)\right)\right),$$

where the sigmoid non-linearity is the standard $\sigma\left(x\right) = \frac{1}{1+\exp^{-x}}$. As before, it is possible to immediately prove the following proposition.

Proposition 4. *Let us consider the sigmoid SF algorithm and let $\mathbf{x}_i$ be points in the original space $\mathbb{R}^M$. Then, the transformations $f_{A1:A4}$, where A2 is now the sigmoid non-linearity, do*

not preserve the structure of the data described either by the Euclidean metric or by the cosine metric.

Proof. We divide this proposition in two parts and we prove each one by counterexample.

Let us focus first on the non-preservation of the Euclidean metric. Let us consider the case in which $\mathbf{x}_1$ is a vector such that $x_{1,j} = 1$, $\forall j$, $1 \leq j \leq M$, $\mathbf{x}_2$ is another vector such that $\mathbf{x}_2 = 2$, $\forall j$, $1 \leq j \leq M$, $L = M$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The Euclidean distance between the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is:

$$D_E\left[\mathbf{x}_1, \mathbf{x}_2\right] = \sqrt{\sum_{j=1}^{M} (1-2)^2} = \sqrt{M}.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{array}{cc} f_{A1}\left(\mathbf{x}_1\right) = \mathbf{I}\mathbf{x}_1 = \mathbf{x}_1 & f_{A1}\left(\mathbf{x}_2\right) = \mathbf{I}\mathbf{x}_2 = \mathbf{x}_2 \\ f_{A2}\left(\mathbf{x}_1\right) = \sigma\left(\mathbf{x}_1\right) = \mathbf{s}_1 & f_{A2}\left(\mathbf{x}_2\right) = \sigma\left(\mathbf{x}_2\right) = \mathbf{s}_2 \\ f_{A3}\left(\mathbf{s}_1\right) = \left[\frac{s_{1,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right] & f_{A3}\left(\mathbf{s}_2\right) = \left[\frac{s_{2,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right] \\ f_{A4}\left(\left[\frac{s_{1,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right]\right) = \left[\frac{1}{\sqrt{N}}\right] = \mathbf{z}_1 & f_{A4}\left(\left[\frac{s_{2,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right]\right) = \left[\frac{1}{\sqrt{N}}\right] = \mathbf{z}_1. \end{array}$$

Thus, $f_{A1:A4}\left(\mathbf{x}_1\right) = \mathbf{z}_1$ and $f_{A1:A4}\left(\mathbf{x}_2\right) = \mathbf{z}_1$. Now, the Euclidean distance between the vectors $f_{A1:A4}\left(\mathbf{x}_1\right)$ and $f_{A1:A4}\left(\mathbf{x}_2\right)$ is:

$$D_E\left[\mathbf{z}_1, \mathbf{z}_2\right] = 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the Euclidean metric. This proves the first part of the proposition.

Let us focus now on the non-preservation of the cosine metric. Let us consider the case in which $\mathbf{x}_1$ is a vector such that $x_{1,j} = 2^j$, $\forall j$, $1 \leq j \leq M$, $\mathbf{x}_2$ is another vector such that $\mathbf{x}_2 = -\mathbf{x}_1$, $L = M = 2$, and $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The cosine distance between the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is:

$$D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] = 1 - \left| \frac{\sum_{j=1}^{M} x_{1,j} x_{2,j}}{\sqrt{\sum_{j=1}^{M} x_{1,j}^2} \sqrt{\sum_{j=1}^{M} x_{2,j}^2}} \right| = 0.$$

Let us now apply the transformation $f_{A1:A4}$ to the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\begin{array}{cc} f_{A1}\left(\mathbf{x}_1\right) = \mathbf{I}\mathbf{x}_1 = \mathbf{x}_1 & f_{A1}\left(\mathbf{x}_2\right) = \mathbf{I}\mathbf{x}_2 = \mathbf{x}_2 \\ f_{A2}\left(\mathbf{x}_1\right) = \sigma\left(\mathbf{x}_1\right) = \mathbf{s}_1 & f_{A2}\left(\mathbf{x}_2\right) = \sigma\left(\mathbf{x}_2\right) = \mathbf{s}_2 \\ f_{A3}\left(\mathbf{s}_1\right) = \left[\frac{s_{1,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right] & f_{A3}\left(\mathbf{s}_2\right) = \left[\frac{s_{2,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right] \\ f_{A4}\left(\left[\frac{s_{1,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right]\right) = \mathbf{z}_1 & f_{A4}\left(\left[\frac{s_{2,j}}{\sqrt{\sum_{i=1}^{N} s_{i,j}}}\right]\right) = \mathbf{z}_2. \end{array}$$

Thus, $f_{A1:A4}\left(\mathbf{x}_1\right) = \mathbf{z}_1$ and $f_{A1:A4}\left(\mathbf{x}_2\right) = \mathbf{z}_2$, where $\mathbf{z}_1 = \begin{bmatrix} 0.70 & 0.71 \end{bmatrix}$ and $\mathbf{z}_2 =$

$\begin{bmatrix} 0.09 & 0.01 \end{bmatrix}$. Now, the cosine distance between the vectors $f_{A1:A4}(\mathbf{x}_1)$ and $f_{A1:A4}(\mathbf{x}_2)$ is:

$$D_C[\mathbf{z}_1, \mathbf{z}_2] \neq 0.$$

Therefore the transformations from A1 to A4 do not preserve the structure of the data described by the cosine metric. This proves the second part of the proposition. ■

In the case of the sigmoid non-linearity, too, the new sigmoid SF cannot preserve either Euclidean distances or cosine distances. Being unable to preserve collinearity (as entailed by Proposition 4), the sigmoid SF cannot preserve the structure of the data defined by cosine neighbourhoodness. As in the case of the ReLU SF, experiments in Section 3.4.2 highlighted this shortcoming by showing that the representation filters defined by sigmoid SF covered large and apparently unstructured portions of the original space.

3.5.1.4 Structure-based sparse filtering

Despite the failure of ReLU SF and sigmoid SF in preserving interesting or useful data structures, we may still expect it to be possible to define alternative forms of SF that can preserve relevant structures.

The best scenario in which to modify the SF algorithm in order to learn new representation filters is the case in which specific a priori knowledge about the shape of the structure of data is available. This a priori information may take the form of knowledge about the structure of a specific data set, such as the data having a concentric structure. In this case it is possible to look for an ad-hoc non-linearity that fits the structure of interest. However, this scenario may be too unrealistic, in that it is very rare to have detailed information about the actual structure of the original data. Alternatively, a priori information may take the form of a more generic knowledge about some behaviour in the structure of the data set, for instance, knowing about the existence of some periodical structure. This sort of a priori knowledge may be more reasonable. In this case as well, it is possible to look for non-linearities that allow the capture of whatever structure may be of interest.

The overall procedure to devise alternative structure-based SF algorithms may then be summarized as: (i) finding a non-linearity that implements filters that fit the structure of the original data; (ii) evaluating and guaranteeing that each step of the new SF algorithm (from A1 to A4) preserves the structure as defined in the original data structure.

It is worth underlining that completing these two steps is far from trivial. It is necessary, at the same time, to find a proper function that meets the desiderata on structure preservation and agrees with the other transformations in SF. Indeed, preservation of structure must not only be satisfied by the specific non-linearity that we choose, but it must also agree with the properties of the remaining steps of SF. In particular, if we keep the steps A3 and A4 unmodified as we suggested above, the normalization steps implicitly set limitations on the types of structures that can be preserved. For instance, even if we were to encode a non-linear function that preserves Euclidean distances, this property would be useless in the overall economy of the SF algorithm, as it would not be preserved by step A4.

Designing customized versions of SF may then be very challenging. Yet it is not impossible, as we will show in Section 4.3 where we develop a novel version of SF specifically designed to address CSA when the data have a periodic structure.

3.5.2 Random projections

Beside algorithms based on the SF prototype, other types of algorithms may be considered as members of the FDL family. In this section we review and interpret the already existing family of random projection algorithms in the light of the framework we developed for FDL algorithms.

RP algorithms (Dasgupta, 2000) are a class of representation learning algorithms for dimensionality reduction, defining the following transformation of the data:

$$\mathbf{Z} = \mathbf{W}\mathbf{X},$$

where $\mathbf{W} \in \mathbb{R}^{L \times M}$, with $\mathbb{R}^L < \mathbb{R}^M$, is a random matrix. RP algorithms are a *sui generis* type of learning algorithms in that no actual learning is happening. Indeed, differently from other algorithms, no optimization of the matrix $\mathbf{W}$ nor of any other parameter takes place. Data are simply projected using a randomly sampled matrix $\mathbf{W}$. This clearly makes the algorithm extremely efficient and simple to use.

There is a deep similarity between SF and RP algorithms. In both cases these algorithms are, at first sight, counter-intuitive and their effectiveness is puzzling. SF performs an optimization on the sparsity of the learned representations, while RP algorithms just perform a purely random projection; neither of them explicitly defines an objective or a constraint about the preservation of relevant information in $\mathbf{X}$. In other words, both algorithms overlook the problem of modelling the pdf that generated the data, and focus only on learning new representations having specific properties, sparsity in the case of SF or simply low dimensionality in the case of RP algorithms.

Under this reading, RP algorithms may then be seen as an instance of a FDL algorithm. However, to better fit in the FDL framework we defined in Section 3.1, it would be interesting to analyse RP algorithms in term of the informativeness and of the infomax principle, as we did for SF.

Concerning the infomax principle, we argue that, analogously to SF, RP algorithms preserve information about the data through the preservation of the structure of the data. It can be proved that random projections can statistically preserve part of the structure of the data: the Johnson-Lindestrauss Lemma (Dasgupta and Gupta, 2003) provides statistical bounds on the preservation of Euclidean distance in the learned representation space and bounds may similarly be provided also for the preservation of cosine distances (Kaski, 1998).

Concerning the informativeness principle, it is more challenging to argue that RP algorithms perform a minimization of the entropy of the learned representations. As no actual learning happens for RPs algorithms, no explicit iterative optimization process takes place. The only

improvement in informativeness may be stated in terms of reducing uncertainty through dimensionality reduction. However, further research and a more rigorous analysis is required to validate this informal statement about decreasing uncertainty through dimensionality reduction.

Interestingly, there is also an important difference between SF and RP algorithms. RP algorithms perform only a linear projection; the simplicity of this transformation is what guarantees the statistical preservation of distances. SF, instead, adds steps of non-linear transformation and normalization; switching from a linear to a non-linear function increases the representational power, but it requires dropping the guarantee of preservation of Euclidean distances.

These similarities and differences between SF and RP algorithms make these two types of algorithms interesting representatives of the family of FDL algorithms. It may be possible to evaluate these two algorithms in light of a trade-off between simplicity and representational power: RP algorithms rely on a simple linear projection and enjoy virtually no training time; SF implements a non-linear transformation and it needs training.

Our study of FDL algorithms is likely not exhaustive and other algorithms may be available in the literature which may fit our FDL paradigm. However, we will stop the review of FDL algorithms here, and, in the next section, we will summarize the results of this chapter.

3.6 Summary of the Chapter

In this section, we summarize the results of this chapter and we highlight the conclusions that we reached.

FDL in information-theoretic terms. At the foundation of our study of FDL lies the understanding that, when learning, unsupervised algorithms must both satisfy an informativeness principle, requiring them to maximize information, and an infomax principle, requiring them to preserve the given information. We then argued that FDL algorithms, too, must conform to this paradigm. In particular, we argued that, despite ignoring the problem of explicitly modelling the true pdf $p(X)$ of the data, FDL algorithms, in order to be successful, *must* somehow preserve information about the distribution of the data $p(X)$. We suggested that this may happen through the use of constraints or priors that allow for information in the original representations $\mathbf{x}_i$ to be carried into the learned representations $\mathbf{z}_i$.

SF in information-theoretic terms. We applied this high-level understanding to analyse the most representative FDL algorithm, that is SF. Through a formal and rigorous analysis we showed that SF is hard-coded with an implicit constraint that guarantees the preservation of a precise form of data structure. This was achieved through a careful analysis of each step of the SF algorithm in which we determined all the properties of the algorithm that are relevant to the preservation of data structures. These properties are now summarized in Table 3.1.

Through a close analysis of these properties we were then able to conclude that our original

Step	*Co-domain*	*Structure preservation*
A1	$\mathbb{R}^L$	× No guarantee on distance preservation ✓Collinearity preservation
A2	With absolute-value: $\mathbb{R}^L_{\geq 0}$ With absolute-value: $\mathbb{R}^L_{>0}$	✓Collinearity preservation
A3	With absolute-value: $\mathbb{R}^L_{\geq 0}$ With absolute-value: $\mathbb{R}^L_{>0}$	✓ Relative distance preservation ✓Collinearity preservation
A4	With absolute-value: $\mathbb{R}^L_{\geq 0}$ With absolute-value: $\mathbb{R}^L_{>0}$	× No distance preservation ✓Collinearity preservation

Table 3.1: Summary of the properties of SF.

thesis was indeed correct: SF satisfies both the informativeness principle, through the proxy of sparsity, and the infomax principle, through the preservation of cosine neighbourhoodness.

SF as representation filters. In our experiments, we were able to validate in a clear way the dynamics and the properties of structure preservation of SF by showing that SF defines precise representation filters in the original space. These dynamics resemble, in part, traditional *sinusoid* or *Walsh filters*, which are often used as bases for sets of coding neurons (Willmore and Tolhurst, 2001). However, differently from these traditional filters which are trained in a DDL framework, SF filters are learned using a FDL paradigm.

SF as a clustering algorithm. We showed that the SF can be seen as an unsupervised soft clustering algorithm based on the cosine metric. This interpretation allowed us to contrast the results of SF with other standard algorithms for clustering based on the Euclidean metric (such soft k-means or GMM). From this perspective SF was shown not to provide a better processing of the data in absolute terms, but instead to provide an alternative interpretation of the data based on a different metric.

Strengths and limits of SF. Our theoretical and empirical study allowed us to explain *why* SF works (by proving its property of preservation of cosine neighbourhoodness) and *when* it should be expected to provide useful representations (by considering the data structure of the samples).

We have been able to highlight the conditions under which SF may be expected to perform significantly better than the standard Euclidean-based alternatives. We showed that whenever the structure of the data with respect to a set of labels can be explained through cosine distances, SF is able to provide cutting-edge performance. Indeed, SF may be seen as an algorithm approximately transforming cosine distances in the original space into Euclidean distances in the representation space; if cosine distances are meaningful with respect to a set of labels, then SF will provide a representation that is especially useful for the large set of standard classifiers that rely on the Euclidean metric in their learning process.

Ideal scenario for SF. The ideal scenario in which to employ SF is one in which relevant information is brought by the radial structure of the data. It is normally assumed

that the data points $\mathbf{x}_i$ are best explained as samples from a multivariate random variable $X = (X_1, X_2, \ldots, X_M)$, where each random variable X_j describes a component $x_{i,j}$. However, given the data points $\mathbf{x}_i$, it is possible to assume that the generating process is better described by a multivariate random variable $X' = \left(X'_1, X'_2, \ldots, X'_{M-1}\right)$, where each random variable describes an angular coordinate θ_j of $\mathbf{x}_i$. SF tries to preserve the information about the $M-1$ angular coordinates θ_j, discarding the information about the radial coordinate ρ. If $p(Y|X)$ is better explained in terms of radial coordinates, then SF is a very reasonable choice for unsupervised representation learning.

This property and behaviour of SF may help explain the success of the application of SF to problems of periocular or iris recognition. Among the several problems to which SF was applied (see Section 2.3.5), a relevant number of papers applies SF to the analysis of ocular images (Raja et al., 2015, 2016b,a; Chhabra and Dutta, 2016; Rattani et al., 2016; Raghavendra and Busch, 2016); it may be hypothesized that this success may be due to the fact that ocular images may easily display a radial symmetry. However, such an hypothesis would need closer study by examining the actual data sets used in each of the studies referenced above.

Bounds for Euclidean distance for SF. However, we also proved that, even if we believe that the structure of the data is better explained in terms of Euclidean distances in Cartesian coordinates, SF can still probabilistically provide good results. This is justified by the fact that, under certain simplified assumptions, the probability that unrelated points with high Euclidean distance will have similar angular coordinates θ_i can be bounded (Section 3.3.7).

Real-world applicability of SF. The practical take-away message from our study of SF was that SF is suitable and very efficient for a specific, although probably limited, set of problems. Whenever the structure of the data is not radial, then SF is reduced to be a probabilistic bet, although a feasible and computationally non-expensive one.

Despite this limitation, we still believe that SF may find large application in real-world scenarios in order to analyse data structures and to process data. While in our experiments we were aware a priori of the metric (either Euclidean or cosine) underlying a synthetic data set, this may not be the case in a real-world scenario. SF, thanks to its scalability and efficiency, could be easily deployed to infer the data structure underlying the data. The usefulness of polar coordinates in several scientific fields and physical applications may suggest that interpreting data according to cosine distance could be a sensible choice. Actually SF could be used to learn a "view" of the data according to the cosine metric. Whenever the underlying structure of the data is unknown, it may also be possible to combine this "cosine view" of the data with a more standard "Euclidean view" of the data. Joining these two different views could provide more informative representations.

Beyond this exploratory approach, a more rigorous approach may be based on using algorithms aimed at analysing the structure of the data (using, for instance, KNN as we did in Section 3.4.5) and then deciding whether to use SF or not.

Development of novel FDL algorithms. The contributions of our study, however, do not stop at SF. Insights and results that we gained studying SF may be extended to other FDL

algorithms more widely.

Being aware of the necessity of complying with the informativeness and the infomax principle is an important guideline for developing novel FDL algorithms, since it would help us to understand what conditions must be satisfied for new algorithms to work successfully.

New SF-like algorithms. We saw that this understanding is particularly precious if we were to develop SF-like algorithms, that is, FDL algorithms that satisfy the informativeness principle through sparsity and that try to satisfy the infomax principle through the preservation of alternative structures underlying the data. On one hand, we showed that alternative data structures may be preserved by designing new SF-like algorithms with different non-linearities; on the other hand, though, we pointed out the importance for such structures to be preserved through all the processing steps of SF. Thanks to this understanding, we could easily analyse alternative SF algorithms relying on standard non-linearities from the neural networks literature and show their inability in preserving data structures of interest. Sigmoid SF or ReLU SF were proposed as potential alternatives to standard SF (Ngiam et al., 2011), but despite their poor empirical performance, no reason was ever given not to expect these algorithms to be potentially useful if properly set or trained. Our simple, yet grounded, theoretical analysis allowed us to conclude that these algorithms may indeed have a very limited use not because of a configuration or a training problem, but because of their inability to preserve data structures of interest.

RP algorithms as FDL algorithms. A formal reading of unsupervised algorithms in terms of the informativeness and infomax principles led us also to reconsider some existing algorithms within the FDL framework. Here we offered the case study of RP algorithms. Like SF, RP algorithms achieve successful performances that may appear, at first sight, unexplainable, but which may be easier to accept within the FDL framework. RP algorithms are a well-understood and widely-studied topic within machine learning, and it may be possible that, by aligning SF and FDL to such a developed area of research, interesting connections and fruitful exchange may follow.

This concludes our analysis of FDL and SF. In the next chapter, we will study how these algorithms may be used to perform CSA.

Chapter 4

Feature Distribution Learning for Covariate Shift Adaptation

This chapter considers the use of FDL algorithms to tackle the problem of covariate shift. It offers a thorough analysis of the possibilities and the limitations of SF and SF-like algorithms in carrying out CSA. Once again, one of the main objectives of this chapter is to provide a clear understanding of how FDL algorithms may perform CSA, when such algorithms may be expected to succeed and how alternative algorithms could be developed.

Section 4.1 provides reasons and justifications for using FDL to tackle covariate shift and articulates which necessary and sufficient conditions a RL algorithm has to meet in order to perform successful CSA. Section 4.2 analyses the SF algorithm in the context of covariate shift and defines in which cases SF is able to perform CSA. Section 4.3 considers how the limitations of SF in the context of covariate shift may be overcome, and it proposes a new algorithm named periodic sparse filtering (PSF). Section 4.4 carries out a formal analysis of the new algorithm with the aim of clearly expressing its strengths and limitations. Section 4.5 validates the theoretical conclusions about performing CSA via SF and PSF using synthetic and real-world data sets. Finally, Section 4.6 synthesizes all the results achieved in this chapter.

4.1 Conceptual Analysis of Covariate Shift Adaptation via Representation Learning

This section offers an introduction to the idea of using FDL for CSA and provides a conceptual analysis of the conditions required for guaranteeing the success of a CSA algorithm. This understanding will inform the ensuing theoretical analysis and the development of SF-like algorithms for CSA.

Section 4.1.1 presents the intuition and the reasons behind the idea of performing CSA via FDL in general, and via SF in particular. Section 4.1.2 discusses the conditions for a RL algorithm to perform successful CSA.

4.1.1 Motivation for performing covariate shift adaptation via feature distribution learning

As discussed in Section 2.4.3.4, several RL algorithms have been proposed for dealing with covariate shift. So far, however, most of these algorithms can be classified as DDL algorithms. To the best of our knowledge, no FDL algorithm has ever been used to perform CSA. We believe that FDL algorithms have never been employed for CSA not for a lack of potential, but mainly because of their recent introduction and the lack of a clear understanding of their dynamics (which we uncovered in Chapter 3).

DDL algorithms are by far the most common representation learning algorithms, and they have been extensively considered for CSA. However, these algorithms are, by definition, particularly vulnerable to covariate shift, as they reconstruct from $\mathbf{X^{tr}}$ the data distribution $p(X^{tr})$, which is inevitably affected by covariate shift. In general, then, DDL algorithms cannot perform CSA, unless in the limit case in which covariate shift is due to a form of noise that may be actually filtered out by the algorithm. As discussed in Section 2.4.3.4, DDL algorithms, as types of RL algorithms, must be enriched with additional objectives or constraints in order to properly perform CSA. For instance, they may be explicitly required during learning to minimize a measure of the distance between the learned distributions $p(Z^{tr})$ and $p(Z^{tst})$. In this way, DDL algorithms can carry out CSA while pursuing their main objective. However, notice that adding a secondary objective comes at the computational cost of making the solution of the optimization problem more difficult and expensive.

FDL algorithms, instead, may bypass the problem of covariate shift. In section 2.2.4 and in Chapter 3 we presented FDL algorithms as learning algorithms that disregard the problem of modelling the distribution of the data $p(X)$ and focus instead only on the problem of shaping a useful distribution of the representations $p(Z)$. In a context in which the data are affected by covariate shift, we believe that the disregard of FDL algorithms for the problem of modelling the original distribution may be particularly useful. If we were to follow the FDL approach and overlook the problem of learning the marginal distribution $p(X^{tr})$, we would not only avoid the computationally hard problem of modelling a marginal pdf from a limited number of samples, but we could circumvent the covariate shift problem. In other words, we could expect a FDL algorithm to be able to "work around" covariate shift by learning representations that are insensitive to the original distance between $p(X^{tr})$ and $p(X^{tst})$. This may happen because, differently from DDL algorithms, FDL algorithms are not necessitated in their definition to suffer from covariate shift thanks to their overlooking the pdf of the data. However, not being necessitated to suffer from covariate shift does not mean that FDL algorithms necessarily perform CSA, either. It is sensible to expect different implementations of FDL algorithms to perform CSA under specific conditions and assumptions.

We then suggest that FDL algorithms may provide a versatile framework in which to define algorithms performing CSA in a simple and straightforward way. As in the i.i.d. scenario, FDL algorithms can disregard the problem of modelling the distribution $p(X)$ and learn a distribution $p(Z)$ with useful properties, so in a covariate shift scenario FDL algorithms can disregard the problem of modelling the distribution $p(X)$ and learn a distribution $p(Z)$ having among its properties, implicitly or explicitly, CSA.

As in Chapter 3 where we highlighted the dependence of the success of a FDL algorithm like SF on its implicit assumptions, so it will be important to understand the conditions under which a FDL algorithm can perform CSA. In order to understand when and how a FDL algorithm can successfully perform CSA, we need to specify what makes a CSA algorithm successful.

4.1.2 Conditions for successful covariate shift adaptation

Performing CSA means compensating for the difference between the distribution of the training data $p(X^{tr})$ and the distribution of the test data $p(X^{tst})$. For the general case of RL, this means learning new representations such that $D[p(X^{tr}), p(X^{tst})] > D[p(Z^{tr}), p(Z^{tst})]$ (see Section 2.4.3.4).

Focusing only on this last condition may be deceiving, as it may lead us to look just for a minimization of a the distance $D[p(Z^{tr}), p(Z^{tst})]$. As in standard FDL, simply maximizing the entropy of the learned representations $H_S[Z]$ would have led us to learn useless representations (see Sections 3.1.2), so in FDL under covariate shift just minimizing the the distance between the learned distributions $D[p(Z^{tr}), p(Z^{tst})]$ would lead us to a meaningless solution. Indeed, once again, a constant function mapping all the samples to an arbitrary representation $\bar{\mathbf{z}}_i$ would perfectly solve the covariate shift problem; all the training data $\mathbf{X^{tr}}$ and test data $\mathbf{X^{tst}}$ would be mapped to $\bar{\mathbf{z}}_i$ and the learned pdfs $p(Z^{tr})$ and $p(Z^{tst})$ would be two identical Dirac delta function, such that $D[p(Z^{tr}), p(Z^{tst})] = 0$. However, as already argued, such representations would be completely useless for any other learning task, as they would lose all the information carried by $p(X)$.

As discussed in Section 2.4.3.1, CSA algorithms are often implemented within supervised systems in order to compensate for a difference in the distribution of training and test samples and thus allow for the learning of a relationship between data and labels. As such, we will restrict our focus to CSA in a supervised scenario. In particular, we will take classification tasks as the reference problem since they are widely considered in the CSA literature and several data sets are available. In this context, a CSA algorithm is required not only to compensate for the difference between the training and test distributions, but also to retain relevant discriminative information. In the case of FDL algorithms like SF, the algorithm has to match the relevant conditional structure of the data.

More formally, the above requirements can be expressed in the following two necessary, but not sufficient, conditions:

Marginal condition a CSA algorithm must compensate for the difference between the marginal distributions of the training and the test data, that is, $D[p(X^{tr}), p(X^{tst})] > D[p(Z^{tr}), p(Z^{tst})]$;

Conditional condition a CSA algorithm must preserve the identity of the conditional distributions, that is, $p(Y|Z) = p(Y|X)$.

These two conditions have to be simultaneously satisfied to achieve good CSA. The first condition is necessary but not sufficient: if an algorithm were not to compensate for the difference

in the distribution of training and test data, then no adaptation would take place; on the other hand, if an algorithm were to compensate for the difference in the distribution of training and test data, then there would still be no guarantee about CSA being successful since all discriminative information may be lost. Similarly, the second condition, too, is necessary but not sufficient: if an algorithm were not to preserve the identity of the conditionals, then ensuing classification would be compromised; if an algorithm were to preserve the identity of the conditionals, then there would still be no guarantee that the distance between the marginals had been reduced.

Notice that these two conditions mirror the assumption of covariate shift. Indeed the statement of covariate shift was divided in two parts (see Section 2.4.2). The first part of the assumption states the problem of the difference between the marginals $p(X^{tr}) \neq p(X^{tst})$; consequently, the first condition requires this problem to be addressed. The second part of the assumption guarantees the identity of the conditionals $p(Y|X^{tr}) = p(Y|X^{tst})$ in order to make generalization possible; consequently, the second condition requires this guarantee to be preserved.

Notice also that the second condition requires only for the preservation of the identity of the conditional distributions. As we are dealing with RL algorithms, a stronger request would be not only to preserve the identity, but also to learn a better-behaved conditional distribution $p(Y|Z)$ that would ease the ensuing classification task. Such a request would be legitimate and within the possibilities of RL, but since we are considering a minimal necessary condition only for CSA, the preservation of the identity is enough.

Now, building on this understanding of CSA, in the next section we will provide a rigorous study of the potential of SF to carry out CSA.

4.2 Theoretical Analysis of Sparse Filtering for Covariate Shift Adaptation

This section analyses the problem of performing CSA via FDL by focusing on SF and studying its strengths and limitations in a covariate shift setting. Considering SF is natural for two reasons: (i) SF is the most emblematic and successful FDL algorithm; (ii) it is possible to rely on the solid theoretical framework developed in Chapter 3 to analyse the ability of SF to carry out CSA.

Following the explanation given in Section 4.1.2, we theoretically analyse *whether* and *under which conditions* SF successfully works for CSA. Section 4.2.1 examines the *marginal condition* for CSA, while Section 4.2.2 considers the *conditional condition* for CSA.

4.2.1 Marginal condition for covariate shift adaptation

In order to prove that SF meets the marginal condition for CSA, it is necessary to show that SF reduces the distance between the marginal distributions of the training and test data. Specifically, it is necessary to show that SF can address both the problem of *domain shift*, that is a difference in the sub-domain of definition of the training and test data $\mathcal{X}^{tr} \neq \mathcal{X}^{tst}$, and the problem of *distribution shift*, that is a difference in the pdfs, $p(X^{tr}) \neq p(X^{tst})$.

Let us start with the domain shift problem. It is easy to show that, through its normalization steps, SF projects all the samples onto a common bounded sub-domain.

Proposition 5. *The sub-domain $\mathcal{Z}$ of the representations $\mathbf{z}_i$ learned by SF is $[0,1]^L$.*

Proof. Let $\mathbf{x}_1 \in \mathcal{X}$ be a generic data sample to be processed through SF. Let the matrix $\tilde{\mathbf{F}}$ be the output of the ℓ_2-normalization along the rows of the SF algorithm, that is:

$$\tilde{\mathbf{F}} = \ell_{2,row}\left(|\mathbf{W}\mathbf{X}|\right).$$

The final output of the SF algorithm is then:

$$\mathbf{z}_1 = \frac{\tilde{f}_{1,j}}{\sqrt{\sum_{j=1}^{L}\left(\tilde{f}_{1,j}\right)^2}}.$$

Every feature $z_{1,j}$ is given by the feature value $\tilde{f}_{1,j}$ normalized by the ℓ_2-norm of $\tilde{\mathbf{f}}_1$. Therefore, it follows that each feature $z_{1,j}$ is bounded within $[0,1]$. Consequently, the representation $\mathbf{z}_1$ is bounded within the hyper-cube $[0,1]^L$. ■

Thus, independently from the original sub-domain of definition of the training and test data, the sub-domain $\mathcal{Z}$ of the learned representations of the training and test data is always identical.

Now let us consider the distribution shift problem. This condition requires the reduction of the distance between the pdfs of the training and the test data with respect to the original distributions: $D\left[p(X^{tr}), p(X^{tst})\right] > D\left[p(Z^{tr}), p(Z^{tst})\right]$. Validating the marginal condition would then require us to consider what the original covariate shift could be and evaluate how it changed in relation to the starting condition. However, instead of studying this relative change, we will evaluate how SF reshapes the pdfs of each learned feature within precise bounds. Formally, this is a looser condition than that explicitly required by the marginal condition expressed above; indeed, in the limit case of a negligible starting covariate shift, reshaping the pdfs within precise bounds is not guaranteed to reduce the covariate shift. However, in the case of sensitive covariate shift, this reshaping is a concrete way to perform CSA. We then analyse how the learned pdf is shaped by studying its first statistical moments.

Proposition 6. *For each learned feature $\mathbf{z}_{\cdot,j}$, the SF algorithm bounds $E[Z_{\cdot,j}] \in [\epsilon, 1]$ and $Var[Z_{\cdot,j}] \in \left[0, 1-\epsilon^2\right]$, where $\epsilon > 0$ is an arbitrarily small value defined in the non-linearity of SF. Moreover, making the assumption that learned representations are $[1,k]$-sparse in population*

and lifetime, and that ϵ is negligible, the bounds can be redefined as $E\left[Z_{\cdot,j}\right] \in \left[\frac{1}{N}, \frac{k}{N}\right]$ *and* $Var\left[Z_{\cdot,j}\right] \in \left[\frac{N-k^2}{N^2}, \frac{Nk-1}{N^2}\right]$.

Proof. The proof of this proposition is based on the following logical steps: (a) re-statement of the basic properties of the learned representations; (b) estimation of the expected value of the distribution of a learned feature (with and without the assumption of k-sparsity); (c) estimation of the second moment of the distribution of a learned feature (with and without the assumption of k-sparsity); (d) estimation of the variance of the distribution of a learned feature (with and without the assumption of k-sparsity).

(a) Let us consider the ℓ_2-normalization steps defining $\tilde{\mathbf{F}}$ and $\mathbf{Z}$. These transformations have two main effects: they constrain all the values in $\tilde{\mathbf{F}}$ and $\mathbf{Z}$ to be within $[0, 1]$; and, they force features or samples to have a square total activation of 1. Formally:

$$0 \leq \tilde{f}_{i,j} \leq 1,\ 0 \leq z_{i,j} \leq 1,\ \ \forall j \in \{1 \dots L\},\ \ \forall i \in \{1 \dots N\},$$

$$\sum_{i=1}^{N} \left(\tilde{f}_{i,j}\right)^2 = 1,\ \ 1 \leq j \leq L$$

$$\sum_{j=1}^{L} \left(z_{i,j}\right)^2 = 1,\ \ 1 \leq i \leq N.$$

(b) Let us now consider a given feature and, for clarity, let us denote this fixed feature as $\bar{j}$ to underline the fact that it is not going to change in the following analysis. We can now analyse the distribution of $\mathbf{z}_{\cdot,\bar{j}}$ by considering its main statistical moments. Let us start by analysing the expected value of the random variable $Z_{\cdot,\bar{j}}$:

$$E\left[Z_{\cdot,\bar{j}}\right] \hat{=} \frac{1}{N} \sum_{i=1}^{N} z_{i,\bar{j}}.$$

After normalizing along the column, the quantity $\sum_{i=1}^{N} z_{i,\bar{j}}$ is not rigidly constrained. The value of the feature $z_{i,\bar{j}}$ can range between ϵ (if the feature $\bar{j}$ happens to be inactive for the sample i) and 1 (if the feature is the only active feature for the sample i). Therefore, the expected value can be bound in:

$$\epsilon \leq E\left[Z_{\cdot,\bar{j}}\right] \leq 1.$$

Let us now make the assumption that the feature $\mathbf{z}_{\cdot,\bar{j}}$ is at most k-sparse, with $1 < k < L$, that is, it is active on a number k of samples, with k greater than 1 and smaller than L. This assumption is justified by considering the properties of population sparsity and lifetime sparsity of SF (Ngiam et al., 2011). In this case, the expected value can be bound in:

$$\frac{1 + (N-1)\epsilon}{N} \leq E\left[Z_{\cdot,\bar{j}}\right] \leq \frac{k + (N-k)\epsilon}{N}.$$

Moreover, if we assume that ϵ is negligible, then the final bound can be re-written as:

$$\frac{1}{N} \leq E\left[Z_{\cdot,\bar{j}}\right] \leq \frac{k}{N}.$$

This proves the first part of our statement.

(c) Let us now consider the estimation of the second moment:

$$M_2\left[Z_{\cdot,\bar{j}}\right] = E\left[\left(Z_{\cdot,\bar{j}}\right)^2\right] \hat{=} \frac{1}{N}\sum_{i=1}^{N} z_{i,\bar{j}}^2.$$

For the same reason we gave above about the admissible values for the feature $\mathbf{z}_{\cdot,\bar{j}}$, the second moment can be bound in:

$$\epsilon^2 \leq M_2\left[Z_{\cdot,\bar{j}}\right] \leq 1.$$

Under the assumption of k-sparsity of $\mathbf{z}_{\cdot,\bar{j}}$, we can get the tighter bounds:

$$\frac{1+(N-1)\epsilon^2}{N} \leq M_2\left[Z_{\cdot,\bar{j}}\right] \leq \frac{k+(N-k)\epsilon^2}{N}.$$

(d) Finally, let us consider the estimation of the variance of $Z_{\cdot,\bar{j}}$:

$$Var\left[Z_{\cdot,\bar{j}}\right] = E\left[Z_{\cdot,\bar{j}}^2\right] - E\left[Z_{\cdot,\bar{j}}\right]^2.$$

Again, using the values we computed for the second moment and the expected value, we can define the following bounds for the variance:

$$\begin{aligned} \epsilon^2 - 1^2 &\leq Var\left[Z_{\cdot,\bar{j}}\right] \leq 1-\epsilon^2 \\ 0 &\leq Var\left[Z_{\cdot,\bar{j}}\right] \leq 1-\epsilon^2. \end{aligned}$$

Under the assumption of k-sparsity we can recompute the bounds:

$$\begin{aligned} Var\left[Z_{\cdot,\bar{j}}\right] &\geq \frac{1+(N-1)\epsilon^2}{N} - \left(\frac{k+(N-k)\epsilon}{N}\right)^2 \\ &\geq \frac{N(1-\epsilon^2+2k\epsilon^2-2k\epsilon)-k^2(1+\epsilon^2-2\epsilon)}{N^2}, \end{aligned}$$

$$\begin{aligned} Var\left[Z_{\cdot,\bar{j}}\right] &\leq \frac{k+(N-k)\epsilon^2}{N} - \left(\frac{1+(N-1)\epsilon}{N}\right)^2 \\ &\leq \frac{N(k-k\epsilon^2+2\epsilon^2-2\epsilon)-1-\epsilon^2+2\epsilon}{N^2}. \end{aligned}$$

If we take ϵ to be negligible, then the bounds of the variance are:

$$\frac{N-k^2}{N^2} \leq Var\left[Z_{\cdot,\bar{j}}\right] \leq \frac{Nk-1}{N^2}.$$

This proves the last part of our statement. ■

Thus, the first part of our statement shows that SF tackles the problem of covariate shift by forcing all the features to have bounded expected values and bounded variances. These bounds on the distribution of the features $Z_{\cdot,j}$ are theoretically independent of the data matrix being processed. Processing $\mathbf{X^{tr}}$ through the SF module returns a new representation $\mathbf{Z^{tr}}$ where each

feature has a distribution within the bounds defined above. If the same SF module were to be used to process the test data $\mathbf{X^{tst}}$ along with the training data $\mathbf{X^{tr}}$, new representation $\mathbf{Z^{tst}}$ would be learned where each feature comes from a distribution within the same bounds. The fact that the new learned distributions $p\left(Z_i^{tr}\right)$ and $p\left(Z_i^{tst}\right)$ have the first statistical moments similarly bounded on the same interval suggests that SF is able, at least in part, to mitigate the problem of covariate shift. More interestingly, the second part of Proposition 6 reveals that, under the assumption of k-sparsity, SF moves the centre of mass of the pdf of each feature $p\left(Z_{\cdot,j}\right)$ towards zero, and it also decreases the variance in proportion to the number of samples N. In other words, SF not only shapes the overall pdf $p(Z)$ towards being mainly localized around zero, but also does the same for the individual pdf of each feature $p\left(Z_{\cdot,j}\right)$. This is consistent with the interpretation of SF in terms of entropy minimization (see Section 3.3.1), according to which the maximization of sparsity is interpreted as a proxy for the minimization of entropy (Principe, 2010; Pastor et al., 2015). Once again, SF can be understood as an algorithm projecting the original data onto representations with a pdf with minimal entropy. Therefore, independently from the original pdfs $p\left(X^{tr}\right)$ and $p\left(X^{tst}\right)$, SF learns a new representation with an entropy-minimized pdf $p(Z)$. Notice, however, that, depending on the structure of the data, the value of k-sparsity may be different when processing training data or test data. In other words, the degree of minimization of the entropy of the learned distributions $p\left(Z^{tr}\right)$ and $p\left(Z^{tst}\right)$ may differ. SF pushes both the learned distributions towards a common entropy-minimized pdf $p(Z)$, but their final distance from $p(Z)$ may vary. The actual value of k-sparsity may then provide an index of the degree CSA provided by SF.

4.2.2 Conditional condition for covariate shift adaptation

Satisfying the marginal condition for CSA is not enough to guarantee that SF always generates useful representations for classification. In order to retain discriminative information, SF must satisfy the requirement of preserving the identity of the conditional distribution $p(Y|Z) = p(Y|X)$.

To prove this condition, we need to determine in which situation SF can preserve the identity of the conditional distributions of the labels given training and test data. We already know from our analysis in Section 3.3.2.5 that SF can preserve the conditional structure of $p(Y|X)$ when this is explained by a metric of cosine neighbourhoodness. By expressing cosine neighbourhoodness in terms of cosine distance, it is immediate to derive the following corollary for the preservation of the structure of the conditional distribution $p(Y|X)$.

Corollary 1. *SF preserves the structure of the conditional distribution $p(Y|X)$ explained by the metric of cosine distance $D_C\left[\mathbf{x}_1, \mathbf{x}_2\right]$.*

SF transforms the conditional distribution $p(Y|X)$ explained by the cosine metric in the original space into a new conditional distribution $p(Y|Z)$ explained by the Euclidean distance in the learned space. Thus, if in the original space a small cosine distance implies a small difference in the conditional distribution:

$$\left[D_C\left[\mathbf{x}_1, \mathbf{x}_2\right] < \epsilon\right] \implies \left[\left|p\left(Y|\mathbf{x}_1\right) - p\left(Y|\mathbf{x}_2\right)\right| < \delta\left(\epsilon\right)\right],$$

where ϵ is an arbitrarily small value $\epsilon > 0$ and $\delta(\epsilon)$ is a function dependent from ϵ and returning an arbitrarily small value $\delta(\epsilon) > 0$, then in the learned space a small Euclidean distance implies a small difference in the conditional distribution:

$$[D_E[\mathbf{z}_1, \mathbf{z}_2] < \epsilon'] \implies [|p(Y|\mathbf{z}_1) - p(Y|\mathbf{z}_2)| < \delta'(\epsilon')].$$

This, in turn, allows standard Euclidean-based classifiers to successfully process the new representations.

In conclusion, bringing together these results for CSA using SF, we have:

Theorem 6. *SF meets the necessary conditions for CSA if the structure of the conditional distribution $p(Y|X)$ is explained by a cosine metric.*

Proof. This theorem follows directly from Propositions 5 and 6, proving the marginal condition, and Corollary 1, proving the conditional condition. ■

SF is then able to perform some degree of CSA under the same conditions required for SF to perform useful unsupervised learning, as discussed in Section 3.3.8. In the next section we will consider how to extend the standard SF algorithm to work and perform CSA under looser conditions.

4.3 Periodic Sparse Filtering

The capacity of SF to perform CSA is limited by the requirement that the data must have a conditional structure explained by the cosine metric. Therefore its ability to perform successful CSA is tied to a specific data structure that a data set is required to exhibit. To overcome this limitation, we propose in this section a new SF-based algorithm that extends this requirement to a conditional distribution explained by a generic periodic structure. This allows us to define a more versatile algorithm that can be expected to work in more scenarios than the standard SF algorithm. We build the new algorithms by starting from the standard SF algorithm and by modifying it according to the guidelines provided in Section 3.5.1.

First of all, we decided to consider periodic structures as we expect them to be useful for modelling many real-world scenarios affected by covariate shift. Indeed, periodic functions would allow us to capture common regularities present in the marginal distributions of training and test data. In the common case of user-dependent data, it is possible to model each user as described by a specific pdf $p\left(X^{(i)}\right)$ on a restricted sub-domain $\mathcal{X}^{(i)}$; labels may then be expected to show some degree of regularity over each sub-domain, such that $p\left(Y|X^{(1)}\right) = p\left(Y|X^{(2)}\right) = \cdots = p\left(Y|X^{(i)}\right)$; learning the periodic behaviour over the set of training users would then allow us to generalize it to the set of test users. For instance, considering once again the case of emotional speech processing (Schuller et al., 2010), samples from speakers in the training set and test set may be defined on different sub-domains; however, the behaviour of the emotional labels may exhibit some form of regularity over the sub-domain of

each speaker.

We set out to define a new SF-based algorithm, which we call *periodic sparse filtering*. To achieve the goal of capturing relevant periodic structures, we enrich the original SF algorithm with two novel crucial properties: (i) the ability to generate periodic filters in the original space, and (ii) the ability to capture the periodic conditional structure underlying the data from available labels.

The first property is necessary to learn a periodic structure. As explained in Section 3.3.4, SF generates hyper-conical filters that can capture a radial structure, but are unable to model more complex periodic structures in the original space $\mathbb{R}^M$. Moreover, from the theoretical study in Section 3.3.6 and from the experiments in Section 3.4.2, we are also aware that the shape of the representation filters in SF is crucially determined by the absolute-value non-linearity. In order to generate periodic filters, we substitute the original non-linearity with a sinusoidal function. The new transformation in PSF is defined as follows:

$$\mathbf{Z} = \ell_{2,col}\left(\ell_{2,row}\left(g\left(\mathbf{W}\mathbf{X}\right)\right)\right),$$

where $g\left(x\right)$ is a positive element-wise sinusoidal function, such as $1+\epsilon+\sin\left(x\right)$ or $1+\epsilon+\cos\left(x\right)$, with $\epsilon > 0$ being an arbitrarily small constant (such as, $\epsilon = 10^{-8}$). Notice that we defined a strictly positive sinusoidal function, that is, a sine or cosine function shifted by $1+\epsilon$ in order to guarantee the strict positivity of the output of $g\left(x\right)$; as in the case of the soft absolute-value non-linearity, strict positivity is required to ensure the correct behaviour of the ensuing normalization steps and to avoid potential division-by-zero errors.

The second property is necessary in order to capture a relevant periodicity underlying the conditional distribution $p\left(Y|X\right)$. Indeed, given a set of unlabelled data, it is possible to discover different periodic functions underlying the data; however, only few of these periodic functions can be usefully related to the conditional distribution $p\left(Y|X\right)$. In order to extract the correct periodic structure, we direct the algorithm to discover the conditional periodic function of interest by using label information. We thus turn the original unsupervised adaptation algorithm into a supervised adaptation algorithm. To do so, we rely on the labels that are provided for the ensuing classification task. Using this additional information, we redefine the learned representations and the loss function as follows. Let $\mathbf{X^{tr}}$ be the training data with its associated labels $\mathbf{Y^{tr}}$. Assuming that we are learning a new representation in $\mathbb{R}^L$, let us partition the L learned features in $C+1$ groups with arbitrary cardinality, corresponding to the C classes defined in $\mathbf{Y^{tr}}$, plus one group for potentially unlabelled samples. We can then re-define the learned representation matrix $\mathbf{Z}$ using the following block matrix notation:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{Z}_{[1,1]} & \mathbf{Z}_{[1,2]} & \dots & \mathbf{Z}_{[1,C]} & \mathbf{Z}_{[1,C+1]} \\ \mathbf{Z}_{[2,1]} & \mathbf{Z}_{[2,2]} & \dots & \mathbf{Z}_{[2,C]} & \mathbf{Z}_{[2,C+1]} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{Z}_{[C,1]} & \mathbf{Z}_{[C,2]} & \dots & \mathbf{Z}_{[C,C]} & \mathbf{Z}_{[C,C+1]} \\ \mathbf{Z}_{[C+1,1]} & \mathbf{Z}_{[C+1,2]} & \dots & \mathbf{Z}_{[C+1,C]} & \mathbf{Z}_{[C+1,C+1]} \end{bmatrix},$$

where $\mathbf{Z}_{[i,j]}$ is the block matrix containing the i^{th} group of learned features from the samples belonging to the j^{th} class. This structure highlights the contribution of each group of learned features to the representation of samples belonging to a given class. Exploiting the representation matrix in this new form, we can redefine the loss function of PSF as:

$$\underset{\mathbf{W}\in\mathbb{R}^{L\times M}}{\operatorname{argmin}} \ \ell_1\left(\mathbf{Z}\right) - \sum_{c=1}^{C} \lambda_c \cdot \ell_1\left(\mathbf{Z}_{[c,c]}\right),$$

where $\lambda_c \in \mathbb{R}$ is a scaling factor. The first term of this new loss function is the same as in SF, and its aim is to push for learning sparse representations. The second term of the loss function is the PSF addition. This term has an opposite effect compared to the first: while the first term tries to reduce and shrink the values of elements of $\mathbf{Z}$, the second term tries to increase the values of the sub-matrices of $\mathbf{Z}$ around the diagonal. Overall, this loss function should push away mass from the components off the diagonal and push it onto the components on the diagonal. These dynamics are reminiscent of learning with energy-based models (LeCun et al., 2006), in which an energy surface over a learned space is pulled down over desired outcomes and pulled up over other outcomes; similarly, even though in a reverse way, our loss function tries to increase the mass over certain outcomes and decrease it over other outcomes. Practically, the learning algorithm is now biased towards generating sparse representations where the c^{th} group of learned features tends to activate for the samples belonging to the c^{th} class.

The pseudo-code for PSF[1] is provided in Algorithm 1; the psuedo-code for the gradient descent on the PSF loss function is provided in Algorithm 2.

Computational complexity. It is worth pointing out that the overall computational complexity of the forward and backward steps of the PSF algorithm is unchanged with respect to the SF algorithm. Indeed, the computational complexity of SF is dominated by the matrix multiplication $\mathbf{WX}$ at step A1. Disregarding optimized implementations of the matrix multiplication algorithm, it is possible to naively take this complexity to be $\mathcal{O}\left(NML\right)$ (Brent and Zimmermann, 2010). Algorithmic differences between SF and PSF take place in step A2 and A5.

In step A2, the original absolute-value non-linearity is substituted by a trigonometric function. In SF, the absolute-value non-linearity has a unitary cost, leading to an overall cost of $NL\mathcal{O}\left(1\right)$ for the application of this function to all the elements of the matrix. In PSF, a trigonometric function, like sine or cosine, has a computational cost of $\mathcal{O}\left(\mathcal{M}\left(n\right)\cdot\log^2\left(n\right)\right)$, where $\mathcal{M}\left(n\right)$ is the computational cost of performing a multiplication and n is the size of the input (Brent and Zimmermann, 2010); the overall cost of the new step A2 in PSF is thus $NL\mathcal{O}\left(\mathcal{M}\left(n\right)\cdot\log^2\left(n\right)\right)$. Notice that this analysis holds also for the back-propagation, as the derivative of sine and cosine is, respectively, a cosine or a sine.

In step A5, new terms are added to the computation of the loss function. SF simply computes a summation over all the components of the matrix $\mathbf{Z}$, which has a time complexity of $NL\mathcal{O}\left(1\right)$. PSF considers additional sums by defining sub-matrices of $\mathbf{Z}$; the complexity of each of these sums is bounded by the complexity of the first sum; thus, the overall cost of the new step A5

[1] The Python source code is available on-line at: https://github.com/FMZennaro/PSF

Algorithm 1 Periodic Sparse Filtering (PSF)

Input: training data $\mathbf{X^{tr}}$; training labels $\mathbf{Y^{tr}}$; target data for adaptation $\mathbf{X^{tgt}}$.
Hyper-parameters: learned dimensionality L; lambda vector λ_i; binary matrix $\mathbf{V}$ defining the block matrix structure of $\mathbf{Z}$ such that $v_{c,j} = 1$ if the j^{th} learned feature is activated by the c^{th} class; gradient descent step η.

$\mathbf{X} \leftarrow \mathbf{X^{tr}} \cup \mathbf{X^{tgt}}$
$\mathbf{W} \leftarrow$ initialize each weight as $\mathcal{N}(0, 1)$
$C \leftarrow$ #classes in $\mathbf{Y^{tr}}$

repeat
  $\mathbf{H} \leftarrow \mathbf{WX}$
  $\mathbf{F} \leftarrow 1 + \epsilon + \sin \mathbf{H}$
  $\tilde{\mathbf{F}} \leftarrow \frac{f_{i,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}}$
  $\mathbf{Z} \leftarrow \frac{\tilde{f}_{i,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}}$
  $\mathcal{L}_1 \leftarrow \sum_{i=1}^{N} \sum_{j=1}^{L} z_{i,j}$
  $\mathcal{L}_2 \leftarrow \sum_{c=1}^{C} \sum_{i:y_i=c} \sum_{j:v_{c,j}=1} \lambda_k \cdot z_{i,j}$
  $\mathcal{L} \leftarrow \mathcal{L}_1 - \mathcal{L}_2$

  $\mathbf{W} \leftarrow \mathbf{W} - \eta \nabla \mathcal{L}$
until termination condition for gradient descent is met
return $\mathbf{Z}$

Algorithm 2 Derivative for PSF

Input: input data $\mathbf{X}$; weight matrix $\mathbf{W}$; PSF output $\mathbf{Z}$; PSF intermediate representations $\mathbf{H}$, $\mathbf{F}$, $\tilde{\mathbf{F}}$.
Hyper-params: lambda vector λ_i.

$\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \leftarrow \frac{\sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2} - z_{i,j} \cdot \sum_{j=1}^{L} \tilde{f}_{i,j}}{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}$
$\left[\frac{\partial \mathbf{Z}}{\partial \mathbf{F}}\right]_{i,j} \leftarrow \frac{\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \sqrt{\sum_{i=1}^{N} f_{i,j}^2} - \tilde{f}_{i,j} \cdot \sum_{i=1}^{N} \left(\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \cdot f_{i,j}\right)}{\sum_{i=1}^{N} f_{i,j}^2}$
$\frac{\partial \mathbf{Z}}{\partial \mathbf{H}} \leftarrow \frac{\partial \mathbf{Z}}{\partial \mathbf{F}} \cdot \cos \mathbf{H}$
$\frac{\partial \mathbf{Z}}{\partial \mathbf{W}} \leftarrow \lambda_i \frac{\partial \mathbf{Z}}{\partial \mathbf{H}} \cdot \mathbf{X}$
return $\frac{\partial \mathbf{Z}}{\partial \mathbf{W}}$

in PSF is unchanged. The complexity of back-propagation is unchanged, as well.

In conclusion, the computational time complexity of the forward and backward steps of SF is the same as PSF. This allows us to state that the novel PSF algorithm is able to keep the efficiency and the simplicity of the original SF algorithm, but, at the same time, it is also able to process data in order to perform CSA under looser conditions than SF.

Having defined a new algorithm, in the following section we will carry out a theoretical analysis of PSF to validate whether and when it can perform CSA.

4.4 Theoretical Analysis of Periodic Sparse Filtering for Covariate Shift Adaptation

This section considers the new SF-based algorithm that we introduced in Section 4.3 and evaluates its strengths and limitations in a covariate shift setting.

By analogy with the theoretical study of SF in Section 4.2, we here analyse the conditions under which PSF successfully works for CSA. Section 4.4.1 examines the *marginal condition* for CSA, while Section 4.4.2 considers the *conditional condition* for CSA.

4.4.1 Marginal condition for covariate shift adaptation

First of all, in order to validate the assertion that PSF can perform successful CSA, we concentrate on the marginal condition which requires the compensation of the distance between the training and the test distribution. As we did in Section 4.2.1, we divide this requirement in two parts: considering first the problem of *domain shift*, $\mathcal{X}^{tr} \neq \mathcal{X}^{tst}$, and then the problem of *distribution shift*, $p\left(X^{tr}\right) \neq p\left(X^{tst}\right)$.

Let us focus first on the problem of domain shift. It is straightforward to show that PSF tackles this problem in the same way as SF. Indeed, PSF maintains the same dynamics of mapping all the training and test data onto a single bounded domain, as made explicit by the following proposition.

Proposition 7. *The sub-domain $\mathcal{Z}$ of the representations $\mathbf{z}_i$ learned by PSF is $[0,1]^L$.*

Proof. The property in this proposition follows from the step of ℓ_2-normalization along the rows, which is exactly the same in both SF and PSF. Therefore, the proof for Proposition 5 holds here as well. ■

Next, let us consider the distribution shift problem. Again, under this respect, PSF behaves in the same way as SF. Indeed, the bounds we defined for SF are not affected by any of the changes we introduced in PSF. We can then state the following proposition.

Proposition 8. *For each learned feature $\mathbf{z}_{\cdot,j}$, the PSF algorithm bounds $E\left[Z_{\cdot,j}\right] \in [\epsilon, 1]$ and $Var\left[Z_{\cdot,j}\right] \in \left[0, 1-\epsilon^2\right]$, where $\epsilon > 0$ is an arbitrarily small value defined in the non-linearity of*

PSF. Moreover, making the assumption that learned representations are $[1, k]$*-sparse in population and lifetime, and that* ϵ *is negligible, the bounds can be redefined as* $E[Z_{\cdot,j}] \in \left[\frac{1}{N}, \frac{k}{N}\right]$ *and* $Var[Z_{\cdot,j}] \in \left[\frac{N-k^2}{N^2}, \frac{Nk-1}{N^2}\right]$.

Proof. Again, these bounds on the distribution of the learned features depend on the ℓ_2-normalization steps, which are the same in SF and PSF. Therefore the proof for Proposition 6 holds here as well. ■

Thus, with respect to the marginal condition for CSA, PSF is able to satisfy this condition in the same way SF does.

4.4.2 Conditional condition for covariate shift adaptation

In considering the conditional condition for CSA we want to evaluate in which cases PSF is able to generate learned representations able to retain useful information for classification. As before, we will analyse this problem by investigating what sort of conditional structure PSF may preserve.

By construction, the PSF algorithm was designed with the idea of preserving a periodic structure underlying the data. It is only natural then to expect that the identity of the conditional distributions, $p(Y|Z) = p(Y|X)$, is preserved when the data exhibit such a structure.

To confirm that our design works as we intended, we first prove a theorem about data structure preservation in PSF. This theorem is the analogue of Theorem 4 about the preservation of cosine neighbourhoodness for SF, and it shows that PSF can preserve a periodic structure.

Theorem 7. *Let* $\mathbf{x}_1 \in \mathbb{R}^M$ *be a point in the original space and let* $\mathbf{z}_1 \in \mathbb{R}^L$ *be its corresponding representation learned by PSF. Then there is an infinite set of points* $\mathbf{x}_i \in \mathbb{R}^M$ *that map onto the same representation* $\mathbf{z}_1$*. The set of the points* $\mathbf{x}_i \in \mathbb{R}^M$ *built from* $\mathbf{x}_1$ *with period* $\mathbf{W}^{-1}\mathbf{k}\pi$*, where* $\mathbf{W}$ *is the weight matrix of PSF and* $\mathbf{k}$ *is a vector of integer constants in* $\mathcal{Z}$*, is included in this set.*

Proof. The proof of this theorem is based on identifying the periodic filters defined by PSF in the original space and showing that points falling within these filters are mapped onto identical representations. The proof makes the following logical steps: (a) rigorous definition of the PSF computation; (b-e) back-computation through all the steps of PSF up to the input (ℓ_2-normalization along the columns, ℓ_2-normalization along the rows, non-linearity, linear projection).

(a) Let us consider two points in the original space $\mathbb{R}^M$:

$$\begin{aligned} \mathbf{x}_1 &= \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,M} \end{bmatrix}^\top \\ \mathbf{x}_2 &= \begin{bmatrix} x_{2,1} & x_{2,2} & \dots & x_{2,M} \end{bmatrix}^\top, \end{aligned}$$

and their corresponding representations in the learned space $\mathbb{R}^L$ defined by PSF:

$$\begin{aligned} \mathbf{z}_1 &= \begin{bmatrix} z_{1,1} & z_{1,2} & \dots & z_{1,L} \end{bmatrix}^\top \\ \mathbf{z}_2 &= \begin{bmatrix} z_{2,1} & z_{2,2} & \dots & z_{2,L} \end{bmatrix}^\top. \end{aligned}$$

Let us also consider a version of PSF implemented using a strictly positive element-wise sine function: $PSF(\mathbf{x}_i) = \ell_{2,col}(\ell_{2,row}(1 + \epsilon + \sin(\mathbf{W}\mathbf{x}_1)))$.

Finally, let us assume that the two learned representations are identical, that is $\mathbf{z}_1 = \mathbf{z}_2$.

(b) By definition of PSF, $\mathbf{z}_1 = \mathbf{z}_2$ implies:

$$\begin{aligned} \ell_{2,col}\left(\tilde{\mathbf{f}}_1\right) &= \ell_{2,col}\left(\tilde{\mathbf{f}}_2\right) \\ \frac{\tilde{f}_{1,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{1,j}^2}} &= \frac{\tilde{f}_{2,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{2,j}^2}}, \end{aligned}$$

where $\tilde{\mathbf{f}}_i = \begin{bmatrix} \tilde{f}_{i,1} & \tilde{f}_{i,2} & \dots & \tilde{f}_{i,L} \end{bmatrix}^\top$ is the intermediate output of PSF defined as $\tilde{\mathbf{F}} = \ell_{2,row}(1 + \epsilon + \sin(\mathbf{W}\mathbf{x}_1))$. Now, for the ℓ_2-normalizations along the columns to be equal, it must hold that:

$$\begin{bmatrix} \frac{\tilde{f}_{1,1}}{d_1} & \frac{\tilde{f}_{1,2}}{d_1} & \dots & \frac{\tilde{f}_{1,L}}{d_1} \end{bmatrix}^\top = \begin{bmatrix} \frac{\tilde{f}_{2,1}}{d_2} & \frac{\tilde{f}_{2,2}}{d_2} & \dots & \frac{\tilde{f}_{2,L}}{d_2} \end{bmatrix}^\top,$$

where $d_i = \sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}$ is a sample-dependent scaling factor. Therefore, it follows that $\mathbf{z}_1 = \mathbf{z}_2$ if and only if $\tilde{\mathbf{f}}_1 = \lambda\tilde{\mathbf{f}}_2$, for $\lambda \in \mathbb{R}$.

(c) By definition of PSF, $\tilde{\mathbf{f}}_1 = \lambda\tilde{\mathbf{f}}_2$ implies:

$$\begin{aligned} \ell_{2,row}(\mathbf{f}_1) &= \lambda\ell_{2,row}(\mathbf{f}_2) \\ \frac{f_{1,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}} &= \lambda\frac{f_{2,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}}, \end{aligned}$$

where $\mathbf{f}_i = \begin{bmatrix} f_{i,1} & f_{i,2} & \dots & f_{i,L} \end{bmatrix}^\top$ is the intermediate output of PSF defined as $\mathbf{F} = 1 + \epsilon + \sin(\mathbf{W}\mathbf{x}_1)$. Now, for the ℓ_2-normalizations along the rows to be equal, it must hold that:

$$\begin{bmatrix} \frac{f_{1,1}}{t_1} & \frac{f_{1,2}}{t_2} & \dots & \frac{f_{1,L}}{t_L} \end{bmatrix}^\top = \lambda\begin{bmatrix} \frac{f_{2,1}}{t_1} & \frac{f_{2,2}}{t_2} & \dots & \frac{f_{2,L}}{t_L} \end{bmatrix}^\top,$$

where $t_j = \sqrt{\sum_{i=1}^{N} f_{i,j}^2}$ is a feature-dependent scaling factor. Therefore, it follows that $\tilde{\mathbf{f}}_1 = \lambda\tilde{\mathbf{f}}_2$ if and only if $\mathbf{f}_1 = \lambda\mathbf{f}_2$.

(d) By definition of PSF, $\mathbf{f}_1 = \lambda\mathbf{f}_2$ implies:

$$1 + \epsilon + \sin(\mathbf{h}_1) = \lambda(1 + \epsilon + \sin(\mathbf{h}_2)),$$

where $\mathbf{h}_i = \begin{bmatrix} h_{i,1} & h_{i,2} & \dots & h_{i,L} \end{bmatrix}^\top$ is the intermediate output of PSF defined as $\mathbf{H} = \mathbf{W}\mathbf{X}$. Now, in order to prove our statement about the set of points $\mathbf{x}_i \in \mathbb{R}^M$ built from $\mathbf{x}_1$ with period $\mathbf{W}^{-1}\mathbf{k}\pi$, let us consider the case where $\lambda = 1$:

$$\begin{aligned} 1 + \epsilon + \sin(\mathbf{h}_1) &= 1 + \epsilon + \sin(\mathbf{h}_2) \\ \sin(\mathbf{h}_1) &= \sin(\mathbf{h}_2). \end{aligned}$$

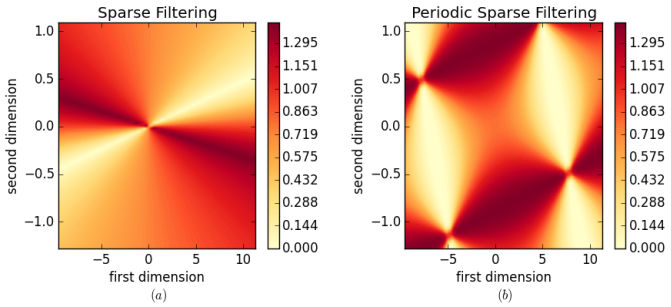

Figure 4.1: Sample filters in the original space $\mathbb{R}^2$ learned by: (a) SF and (b) PSF

For the applications of the sinusoidal function to be equal, it must hold that:

$$\begin{aligned} \sin(\mathbf{h}_1) &= \sin(\mathbf{h}_2) \\ \mathbf{h}_1 &= \arcsin(\sin(\mathbf{h}_2)) \\ \mathbf{h}_1 &= \mathbf{h}_2 + \mathbf{k}\pi, \end{aligned}$$

where $\mathbf{k}$ is a vector of feature-dependent periodic factors in $\mathcal{Z}$.

(e) By definition of PSF, $\mathbf{h}_1 = \mathbf{h}_2 + \mathbf{k}\pi$ implies:

$$\begin{aligned} \mathbf{W}\mathbf{x}_1 &= \mathbf{W}\mathbf{x}_2 + \mathbf{k}\pi \\ \mathbf{x}_1 &= \mathbf{x}_2 + \mathbf{W}^{-1}\mathbf{k}\pi. \end{aligned}$$

Thus, there are infinite points $\mathbf{x}_i \in \mathbb{R}^M$ built from $\mathbf{x}_1$ with period $\mathbf{W}^{-1}\mathbf{k}\pi$ that maps onto the same representation $\mathbf{z}_1$. ■

This theorem proves that the PSF algorithm can define a specific frequency for each dimension in the original space; all the points with coordinates that are multiples of these frequencies are then mapped onto the same representation. Notice that this set of points has been defined by assuming the value of the multiplicative constant $\lambda = 1$; different values of λ may define other sets of points that are mapped onto the same representation.

Practically, a sinusoidal non-linearity allows for the generation of periodic filters that can regularly tile the whole original space. This behaviour can be easily illustrated using representation filters (as explained in Section 3.3.4). Figure 4.1 provides a comparison between the representation filters of standard SF and the new periodic filters of PSF; the periodic filters learned by PSF appear to be versatile and they may be shaped into different forms, ranging from parallel stripes with a chosen orientation to squares tiling the whole space.

Now, from Theorem 7 we can immediately derive the following corollary on the preservation of the structure of the conditional distribution $p(Y|X)$.

Corollary 2. *PSF preserves the structure of the conditional distribution $p(Y|X)$ explained by a periodic metric $D_P[\mathbf{x}_1, \mathbf{x}_2] = \ell_p(g_{\mathbf{k}}(\mathbf{x}_1) - g_{\mathbf{k}}(\mathbf{x}_2))$, where $g_{\mathbf{k}}(\mathbf{x}_i)$ is an element-wise periodic function with periods $\mathbf{k}$ and $\ell_p(\cdot)$ is an ℓ_p-norm.*

According to this corollary, if in the original space a small periodic distance implies a small difference in the conditional distribution:

$$[D_P[\mathbf{x}_1, \mathbf{x}_2] < \epsilon] \implies [|p(Y|\mathbf{x}_1) - p(Y|\mathbf{x}_2)| < \delta(\epsilon)],$$

then, in the learned space, a small Euclidean distance implies a small difference in the conditional distribution:

$$[D_E[\mathbf{z}_1, \mathbf{z}_2] < \epsilon'] \implies [|p(Y|\mathbf{z}_1) - p(Y|\mathbf{z}_2)| < \delta'(\epsilon')].$$

In conclusion, putting together these results for CSA using SF, we have:

Theorem 8. *PSF meets the necessary conditions for CSA if the structure of the conditional distribution $p(Y|X)$ is explained by a periodic metric.*

Proof. This theorem follows directly from Propositions 7 and 8, proving the marginal condition, and Theorem 7 and Corollary 2, proving the conditional condition. ■

PSF can then perform CSA under the looser requirement of a periodic structure. However this increased capacity requires additional information for directing the learning process; since many different periodic structures may be learned, PSF needs to rely on side information in the form of labels in order to extract a useful or meaningful structure.

In the next section we will evaluate empirically the SF and PSF algorithm and compare them to other algorithms from the literature.

4.5 Experimental Validation of Covariate Shift Adaptation via Sparse Filtering and Periodic Sparse Filtering

In this section we validate and test our theoretical results on the use of FDL algorithms for CSA. We start by running experiments on synthetic data sets in order to obtain a better understanding and to be able to easily visualize the effects of covariate shift and the contributions of CSA. We then execute a series of experiments on real data sets in which we measure the effectiveness of SF and PSF against other CSA algorithms used in the machine learning literature.

Section 4.5.1 begins by validating the conditions proposed above for successful CSA for both SF and PSF. Section 4.5.2 provides a clear and illustrative comparison of different CSA algorithms on carefully designed data sets. Section 4.5.3 extends the previous comparison from synthetic to real-world data showing the place and the usefulness of PSF.

4.5.1 Properties of sparse filtering and periodic sparse filtering in performing covariate shift adaptation

We begin by running simple simulations on crafted toy data sets in order to show the relevance of the conditions for successful CSA that we outlined in Section 4.1.2. These simulations aim to verify: (i) the dependence of the performance of SF on the satisfaction of its marginal and conditional conditions for successful CSA (see Section 4.2.1 and 4.2.2); and, the dependence of the performance of PSF on the satisfaction of its marginal and conditional conditions for successful CSA (see Section 4.4.1 and 4.4.2).

Data set. We generate a simple synthetic data set with the following requirements: (i) data must be easily visualizable; (ii) covariate shift must affect at least one dimension of the input; (iii) the labelling function must be a periodic function defined on the dimension exhibiting the covariate shift; (iv) the effect of covariate shift must be evident if we process data under the i.i.d. assumption. These requirements allow us to model in a simplified way the case of user-dependent data, in which data from different users have distributions located far apart in the feature space but exhibit local regularities with respect to the labelling function.

We then generated a data set by sampling points from two bivariate Gaussian pdfs. The training data set $\mathbf{X^{tr}}$ is made up by 50 samples from $\mathbf{X^{tr}} \sim \mathcal{N}\left(\begin{bmatrix} 2\pi \\ 2 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & .5 \end{bmatrix}\right)$, while the test data set $\mathbf{X^{tst}}$ is made up by 50 samples from $\mathbf{X^{tst}} \sim \mathcal{N}\left(\begin{bmatrix} -2\pi \\ 2 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & .5 \end{bmatrix}\right)$. Binary labels over the training data and the test data are defined by a deterministic square wave function with period 1 on the domain of the first feature. Figure 4.2 shows the synthetic data in two dimensions, while Figure 4.3 shows the projection of the synthetic data along the first dimension, that is, the dimension affected by covariate shift.

Experimental protocol. We train a standard SF module and a PSF module to learn the representations $\mathbf{Z^{tr}}$ and $\mathbf{Z^{tst}}$. We set the dimensionality of the learned representations to $L = 2$ for visualization reasons. In the case of PSF, we set the scaling factors λ to 1 to keep all the loss terms in the same order of magnitude. We run a classification task using standard algorithms, linear SVM and RBF SVM, on both the original data ($\mathbf{X^{tr}}$ and $\mathbf{X^{tst}}$) and the processed data ($\mathbf{Z^{tr}}$ and $\mathbf{Z^{tst}}$). The linear SVM uses a fixed penalty $C = 1$; the RBF SVM also uses a fixed penalty $C = 1$ and a fixed $\gamma = \frac{1}{M} = 0.5$.

We analyse the results as follows: (i) we plot example filters learned by SF and PSF; (ii) we use a Kolmogorov-Smirnov (KS) test to analyse the distribution of each feature (see Section 2.4.2); (iii) we provide the classification accuracy on the training and the test data; (iv) we compute an estimation of cross-domain generalization using the metric of percentage drop (PD) (see Section 2.4.3).

Results. Figure 4.4 shows the actual filters instantiated by SF and PSF in relation to the synthetic data. The figure shows the ability of PSF to perform periodic filtering and it confirms that PSF is able, after successful learning, to instantiate filters that better conform to the data.

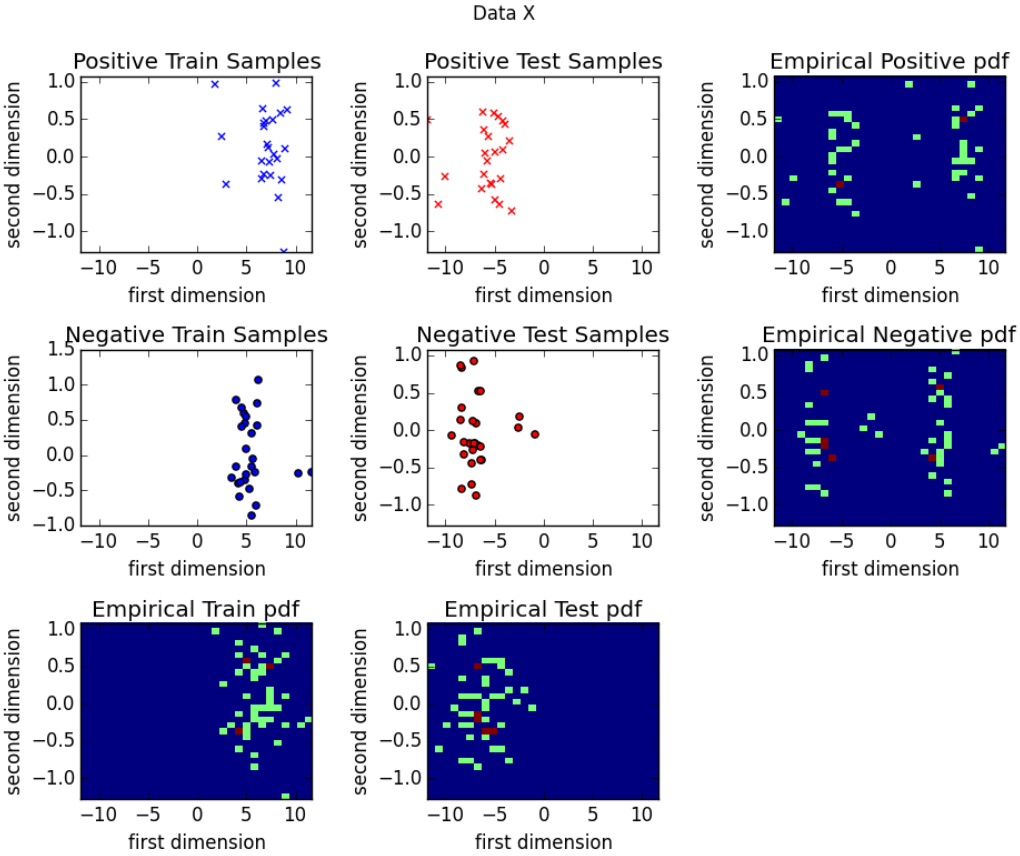

Figure 4.2: Synthetic data for validating CSA properties of SF and PSF.
For reasons of space and clarity, we avoided labelling each sub-graph. We used the blue colour for training samples and the red colour for test samples; we used crosses for positive-labelled samples and circles for negative-labelled samples.
The first column illustrates the training samples in $\mathbb{R}^2$ and their empirical distribution as a two-dimensional histogram; the second column illustrates the test samples in $\mathbb{R}^2$ and their empirical distribution as a two-dimensional histogram.
The first row illustrates the positive samples in $\mathbb{R}^2$ and their empirical distribution as a two-dimensional histogram; the second row illustrates the negative samples in $\mathbb{R}^2$ and their empirical distribution as a two-dimensional histogram.
Looking at the third row and the third column it is immediate to verify that the conditions of the covariate shift assumption are met. Indeed, observing the third row, it is immediate to spot the covariate shift occurring along the first dimension between the training and the test data, $p\left(X^{tr}\right) \neq p\left(X^{tst}\right)$. Observing the third column, instead, it is immediate to notice that the conditional distribution is the same over training and test data, $p\left(Y|X^{tr}\right) = p\left(Y|X^{tst}\right)$.

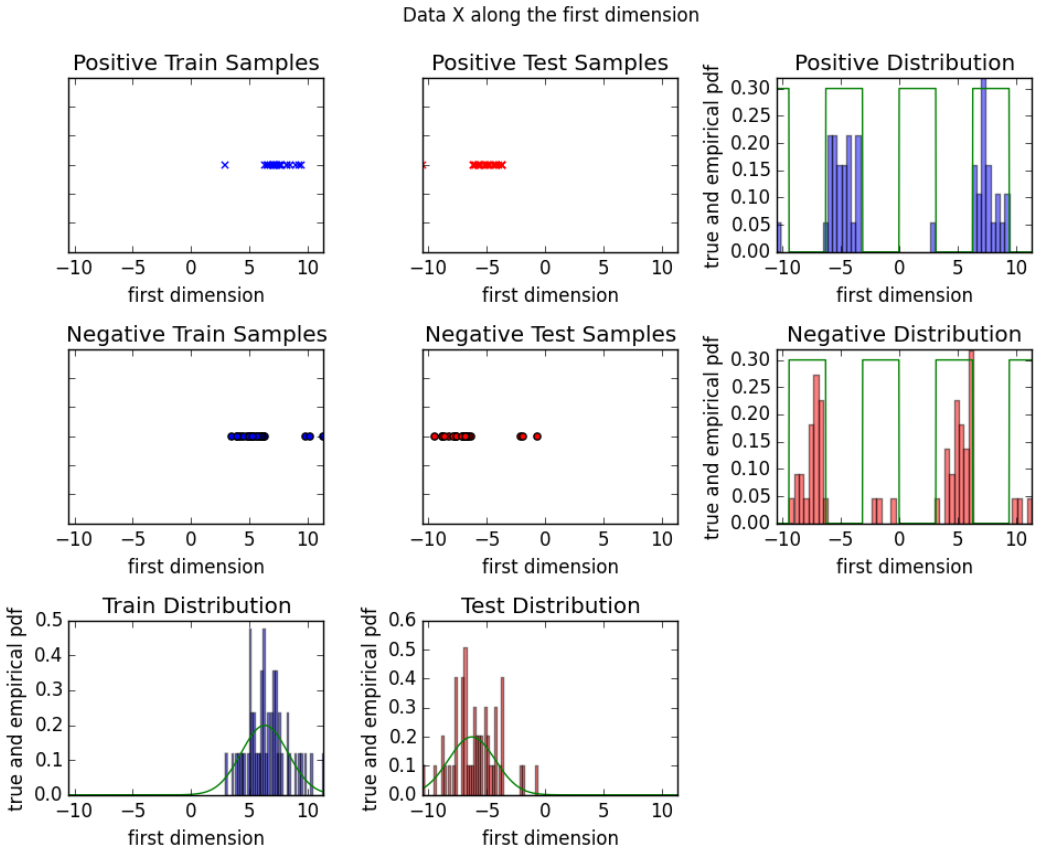

Figure 4.3: Projection of the synthetic data for validating CSA properties of SF and PSF along the first dimension.

As covariate shift happens on the first dimension, we provide a closer look at this dimension through a projection of the data. Colour and symbol conventions are the same as in Figure 4.3. The first column illustrates the training samples in $\mathbb{R}$ and their empirical distribution as a histogram plotted over the true distribution (in green); the second column illustrates the test samples in $\mathbb{R}$ and their empirical distribution as a histogram plotted over the true distribution (in green).

The first row illustrates the positive samples in $\mathbb{R}$ and their empirical distribution as a histogram plotted over the true distribution (in green); the second row illustrates the negative samples in $\mathbb{R}$ and their empirical distribution as a histogram plotted over the true distribution (in green).

As for Figure 4.2, this figure allows us to confirm that covariate shift happens along the first dimension. In particular, from the third row we infer the inequality of the marginal distributions, $p\left(X^{tr}\right) \neq p\left(X^{tst}\right)$, while from the third column we infer the equality of the conditional distributions, $p\left(Y|X^{tr}\right) = p\left(Y|X^{tst}\right)$.

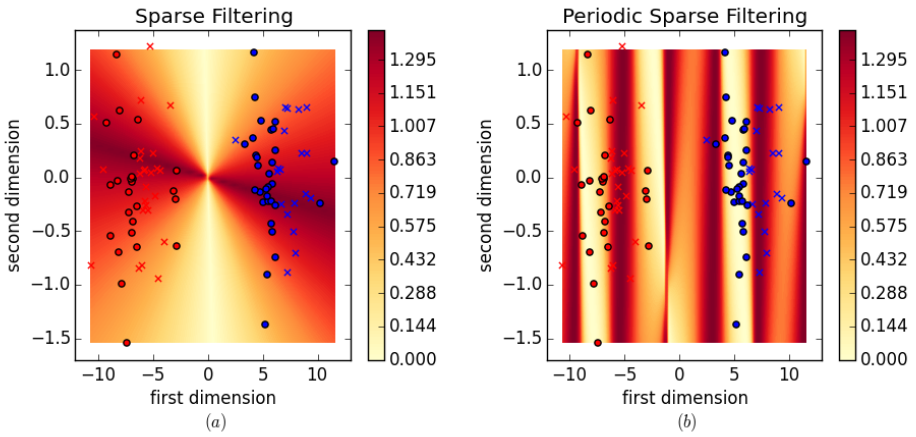

Figure 4.4: (a) Illustration of a filter instantiated by SF. (b) Illustration of a filter instantiated by PSF.

	Kolmogorov-Smirnov	
Data	(train,test)	(positive,negative)
Raw	0.5	0.5
SF	0.0	0.0
PSF	0.4 ± 0.49	0.5 ± 0.5

Table 4.1: Kolmogorov-Smirnov statistical test (p-value 0.05) on the synthetic data set before and after CSA.
We test two hypotheses: (i) that the features for training data and test data come from different distributions, and (ii) that the features for positive-labelled data and negative-labelled data come from different distributions. Averages and standard deviations are computed over 10 simulations with randomly re-sampled data.

Table 4.1 contains the result of the KS test on synthetic data. When using raw data, the KS test easily detects covariate shift on one dimension (the first dimension) and a difference in the distribution of positive-labelled and negative-labelled data on one dimension (again the first dimension). After processing using the SF module, all the differences are lost. The KS test reveals that CSA took place, in that the distributions of the training data and the test data now appear to be identical. However, the KS test demonstrates also that, unfortunately, any difference in the distribution of positive-labelled and negative-labelled data is lost. This is consistent with our theoretical understanding: SF satisfies the marginal condition for successful CSA, but since the structure of the data is not radial, it cannot meet the conditional condition. When using the PSF-processed data, the results exhibit a much higher variance due to the learning process of extracting a periodic structure from the data. The KS test suggests that some degree of CSA can take place and, at the same time, some degree of information discriminating between the distribution of positive-labelled and negative-labelled can be retained. Again, this is consistent with our understanding of CSA, in that PSF can, in this scenario, meet both the marginal and the conditional condition for successful CSA.

	Linear SVM			RBF SVM		
Data	acc(tr)	acc(te)	PD	acc(tr)	acc(te)	PD
Raw	0.78 ± 0.08	0.51 ± 0.06	35.06 ± 4.47	0.97 ± 0.02	0.51 ± 0.06	48.83 ± 5.86
SF	0.61 ± 0.03	0.52 ± 0.06	8.75 ± 11.07	0.59 ± 0.04	0.53 ± 0.06	9.2 ± 14.19
PSF	0.726 ± 0.12	0.69 ± 0.15	6.58 ± 10.65	0.72 ± 0.12	0.69 ± 0.16	5.82 ± 10.6

Table 4.2: Classification accuracy on the synthetic data set. We computed the classification accuracy on the training data set, the classification accuracy on the test data set, and the relative percentage drop, both when using a linear SVM classifier and a RBF SVM classifier. Averages and standard deviations are computed over 10 simulations with randomly re-sampled data.

Table 4.2 contains the results of classification on synthetic data. When using raw data, the linear SVM achieves low performance because of the impossibility of separating the data linearly; on the training set itself the accuracy is low, and on the test set it does not improve over the chance level. On the other hand, the RBF SVM reaches almost perfect discrimination on the training data, being able to perfectly separate the data in a high-dimensional space; however, it also suffers a severe drop when applied to the test data, falling back to a chance-level performance. The percentage drop for both the classifiers is consequently very high. After processing using the SF module, the classification results worsen for both the linear SVM and the RBF SVM. The discriminative information in the new representations is so low that the linear SVM and the RBF SVM fail at classifying not only the test data, but also the training data. The percentage drop is reduced, but this is mainly due to the dramatic drop in the accuracy on the training data. This result is coherent with our theoretical study: by not satisfying the conditional condition for successful CSA, SF compromises the possibility of a good classification. When using the PSF-processed data, a clear improvement can be noticed. The classification accuracy on the training data set is lower than in the case of raw data, but this performance is not indicative of generalization. On the other hand, the classification accuracy on the test data set is significantly higher than the chance level and higher than when using raw data or SF-processed data. The percentage drop is sensibly reduced, both because of a limited drop in performance on the training data, but, more importantly, because of a clear increase in performance on the test data. As predicted, PSF was able to meet all the conditions for successful CSA and was therefore able to learn a good representation for the data that led to an improved classification accuracy.

These results confirm, then, the importance of satisfying all the conditions for successful CSA in order to learn useful representations for classification. The KS test shows that both SF and PSF can achieve some adaptation, thus meeting the marginal condition for successful CSA. However, the same test also highlights that only PSF preserves discriminative information, thus meeting the conditional condition for successful CSA. Consequently, when using representations learned through SF the classification accuracy is severely compromised; only when using representations learned through PSF does the classification accuracy actually improve.

In general, it is also important to point out that the higher flexibility and learning capacity of PSF comes at the cost of making the learning problem more challenging. This is shown by the high variance of the results obtained by PSF, suggesting that the algorithm may be sensible

to its initialization and that some simulations may lead to very good solutions while other simulations may learn less useful filters. The result from the KS test may be used as an index to get a hint on which instantiations of PSF learned good filters and are likely to contribute to an improvement of the overall performance. Also, PSF performance may be further refined by fine-tuning the hyper-parameters (L, λ) through proper model selection via cross-validation.

4.5.2 Comparison of sparse filtering and periodic sparse filtering against other covariate shift adaptation algorithms

In this section, we extend the results of the previous simulations to a wider range of CSA algorithms. The aim of these experiments are: (i) further confirming the ability of SF and PSF to reduce the distance between training and test data on more complex data sets; (ii) comparing SF and PSF against other well-known algorithms from the machine learning literature; (iii) highlighting that the success of these algorithms largely depends on whether the conditional condition for successful CSA is met by the data.

In this simulation we consider five different CSA algorithms: SF, PSF, IW, SSA and DAE (see Sections 2.4.3.5 and 2.4.3.4 for details on IW, SSA and DAE). Beyond our algorithms (SF and PSF), the other algorithms were chosen for the following reasons. (i) IW, SSA and DAE are well-known algorithms in the CSA community and they have been used to tackle covariate shift in several different scenarios (see Sections 2.4.3.5 and 2.4.3.4). (ii) IW and SSA have a solid theoretical backing that allows us to analyse and interpret their results. (iii) DAE was shown to achieve state-of-the-art performances in several applications and, thus, it provides a competitive benchmark against which we can evaluate SF and PSF.

In considering the marginal condition for successful CSA, we show once again that SF and PSF meet this requirement; for the other algorithms from the literature we take for granted that they can perform some degree of CSA and can therefore meet the marginal condition for successful CSA. Concerning the conditional condition for successful CSA, its satisfaction cannot be taken for granted in the same way, as it depends on the specific data to which we apply the CSA algorithms. In the case of four out of five of the CSA algorithms that we considered (SF, PSF, IW, SSA), the conditional condition for successful CSA can be made explicit; for these algorithms we will design specific data sets meeting these conditions in order to study the difference in performance when the assumptions are met and when they are not. In one case (DAE), the conditional condition for successful CSA cannot be expressed in explicit terms; this limits the possibility of crafting a data set specifically designed for the DAE algorithm, but it will not hinder our ability to compare it against the other CSA algorithms.

Data Set. With reference to the four CSA algorithms for which we can express conditional conditions for successful CSA, we generated four different data sets:

- *Radial data set:* the training data set $\mathbf{X^{tr}}$ consists of 500 samples from the distribution $X^{tr} \sim \mathcal{N}\left(\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 500 samples from the distribution $X^{tst} \sim \mathcal{N}\left(\begin{bmatrix} -0.5 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tgt}}$ consists of 250 samples from $p\left(X^{tst}\right)$. $p\left(Y|X\right)$ is described by the deterministic function $f\left(\mathbf{x}_i\right) = \begin{cases} 1 & \text{if } |x_{i,1}| > |x_{i,2}| \\ 0 & \text{otherwise} \end{cases}$, which defines two cones centred around the x-axis.

- *Periodic data set:* the training data set $\mathbf{X^{tr}}$ consists of 500 samples from the distribution $X^{tr} \sim \mathcal{N}\left(\begin{bmatrix} 2\pi \\ 0 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 500 samples from the distribution $X^{tst} \sim \mathcal{N}\left(\begin{bmatrix} -2\pi \\ 0 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tgt}}$ consists of 250 samples from $p\left(X^{tst}\right)$. $p\left(Y|X\right)$ is described by the deterministic function $f\left(\mathbf{x}_i\right) = \begin{cases} 1 & \text{if } \sin\left(|x_{i,1}|\right) > 0 \\ 0 & \text{otherwise} \end{cases}$, which defines a periodic pattern perpendicular to the x-axis.

- *Smooth data set:* the training data set $\mathbf{X^{tr}}$ consists of 250 samples from the distribution $X^{tr1} \sim \mathcal{N}\left(\begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\right)$ and 250 samples from the distribution $X^{tr2} \sim \mathcal{N}\left(\begin{bmatrix} -2 \\ 3 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 250 samples from the distribution $X^{te1} \sim \mathcal{N}\left(\begin{bmatrix} 3 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$ and 250 samples from the distribution $X^{te2} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$; $\mathbf{X^{tgt}}$ consists of 125 samples from $p\left(X^{te1}\right)$ and 125 samples from $p\left(X^{te2}\right)$. $p\left(Y|X\right)$ is described by the deterministic function $f\left(\mathbf{x}_i\right) = \begin{cases} 1 & \text{if } \frac{1+\tanh(x_{i,1}+\min(0,x_{i,2}))}{2} > 0.5 \\ 0 & \text{otherwise} \end{cases}$.

- *Diagonal data set:* the training data set $\mathbf{X^{tr}}$ consists of 500 noisy samples taken along the diagonal of the first quadrant, where the first component $x_{i,1}$ is sampled from $\mathcal{U}\left(0,3\right)$ and the second component is $x_{i,2} = x_{i,1} + \epsilon$, where $\epsilon \sim \mathcal{N}\left(0, 0.2\right)$; $\mathbf{X^{tst}}$ consists of 500 noisy samples taken along the diagonal of the fourth quadrant, where the first component $x_{i,1}$ is sampled from $\mathcal{U}\left(-3,0\right)$ and the second component is $x_{i,2} = x_{i,1} + \epsilon$, where again $\epsilon \sim \mathcal{N}\left(0, 0.2\right)$; $\mathbf{X^{tgt}}$ consists of 250 noisy samples taken along the diagonal of the fourth quadrant as $\mathbf{X^{tst}}$. $p\left(Y|X\right)$ is described by the deterministic function $f\left(\mathbf{x}_i\right) = \begin{cases} 1 & \text{if } |x_{i,2}| > 1.5 \\ 0 & \text{otherwise} \end{cases}$.

Each data set has been designed to meet the assumptions of a particular CSA algorithm. The *radial* data set meets the requirement of a radial structure required by SF. The *periodic* exhibits a periodic data set structure that fits the assumptions of PSF. The *smooth* data set satisfies the assumption of IW (see Section 2.4.3.5) by defining a continuous and well-behaved conditional distribution over the training and the test domain; notice that this data set is the

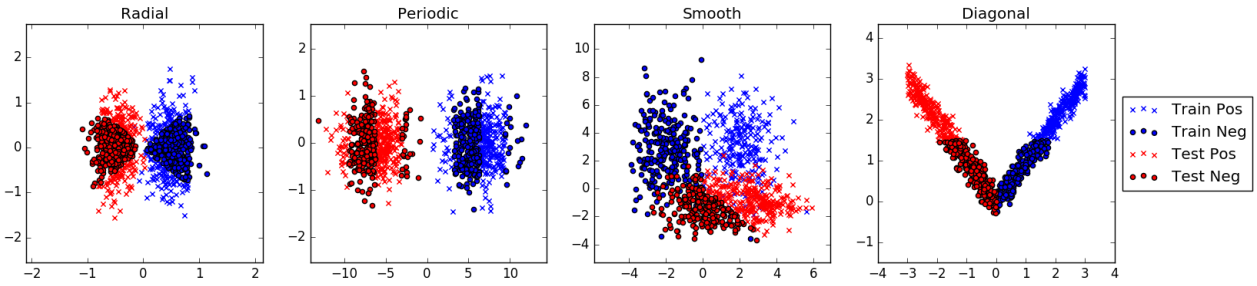

Figure 4.5: Synthetic data sets for evaluating CSA with different CSA classification systems. The *radial*, *periodic*, *smooth* and *diagonal* data sets are built respectively on the CSA assumptions of SF, PSF, IW and SSA.

same as the one generated by Hachiya et al. (2012) and used to illustrate the effectiveness of IW. Finally, the *diagonal* data set meets the assumptions of SSA (see Section 2.4.3.5) by generating a training set and a test set having PCA spaces easily projectable on each other. Figure 4.5 illustrates the four data sets.

Experimental protocol. In order to highlight the dependency of CSA algorithms on their underlying assumptions about the structure of the data and to measure SF and PSF against state-of-the-art methods, we implemented the following classification systems:

(i) *SVM (without CSA)*: this model does not perform any CSA and it is used only to provide a baseline against which to evaluate the contribution of CSA algorithms.

The linear SVM is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$, using a fixed penalty $C = 1$.

(ii) *SF+SVM*: this model performs CSA via SF and classification using a linear SVM.

The SF module is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ for 500 iterations, the learned dimensionality is set to $L = 2$. The linear SVM is trained as for system (i).

(iii) *PSF+SVM*: this model performs CSA via PSF and classification using a linear SVM.

The PSF module is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}, \mathbf{Y^{tr}}\}$ for 500 iterations, the learned dimensionality is set to $L = 2$ (divided evenly between the two classes), the chosen non-linearity is the positive sine, and $\lambda = 1.0$ to balance the loss terms. The linear SVM is trained as for system (i).

(iv) *IW+LSPC*: this model performs CSA using IW with commonly used settings for CSA based on Hachiya et al. (2012) and it performs classification using a least-square probabilistic classifier (LSPC) (Sugiyama et al., 2012).

The IW algorithm runs on $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ using the pre-set *uLSIF* algorithm with 250 bases, and looking for optimal σ in the candidate set $\{0.1, 0.2, 0.5, 1, 2, 3\}$ and for optimal λ in the candidate set $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10\}$. LSPC is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ with the same pre-configured candidate sets $\sigma = \{0.1, 0.2, 0.5, 1, 2, 3\}$ and $\lambda = \{0.1, 0.17, 0.32, 0.56, 1\}$.

(v) *SSA+SVM*: this model performs CSA via SSA and classification using a linear SVM.

The SSA module is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ using 2 PCA components. The linear SVM is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$, using a penalty $C = 3$ to achieve the same baseline results as systems (i)-(iii).

(vi) *DAE+SVM*: this model performs CSA via DAE and classification using a linear SVM.

The DAE module is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tgt}}\}$ for 10000 epochs with mini-batches of size 50 and learning rate of 0.001; we set the learned dimensionality to $L = 1$, the non-linearity to a sigmoid and the noise to Gaussian $\mathcal{N}(0, 0.1)$. The linear SVM is trained as for system (i).

For details on the implementation of each classification system, see Appendix A.

To validate once again that SF and PSF meet the marginal condition required for CSA, we estimate the distance between the pdfs of the training and the test data before and after CSA by computing the MMD distance (with an automatically estimated hyper-parameter σ), and we report the percentage difference in the distance.

To validate that the effectiveness of CSA algorithms for classification depends on the satisfaction of their specific conditional conditions for successful CSA, we run a classification task and we evaluate the percentage difference in accuracy with and without CSA. Using a relative measure allows us to account for the implementation differences between the CSA systems (such as in the use of labels for adaptation, type of classifiers and implementation details).

Results. Table 4.3 lists the percentage differences in the MMD distance between the marginal pdfs of the training and test data following the adoption of CSA. These results confirm that both SF and PSF are able to significantly reduce the distance between the marginal pdfs of the training and the test data, thus satisfying the marginal condition for successful CSA.

Table 4.4 shows the raw accuracy before and after introducing a CSA module. The results show that, even if the baselines of the five CSA classification systems are not exactly the same, they are still comparable; therefore, it makes sense to contrast the percentage change in accuracy when processing a given data set.

Table 4.5 reports the percentage difference in accuracy for all the CSA systems on the four data sets. These results confirm a direct correlation between the success in classification and the satisfaction of the conditional condition for successful CSA related to the assumptions of each algorithm. Indeed, the most relevant positive results are displayed on the main diagonal of Table 4.5, corresponding to the cases when a CSA classification system is used on data matching its assumptions; that is, the best results are obtained when applying SF to the *radial* data set, PSF to the *periodic* data set, IW to the *smooth* data set and SSA to the *diagonal* data set. Off the diagonal, the violation of the assumptions results in very limited improvement or even a decrease in classification accuracy.

For instance, SF yields a negligible increase in accuracy when applied to the *periodic* data set, due to the fact that classification with a linear SVM remains basically at a random guess level before and after CSA, but it causes a severe drop in accuracy when applied to other data sets, due to the assumption mismatch.

	SF	***PSF***
Radial	$-100.1\% \pm 0.04\%$	$-50.5 \pm 10.2\%$
Periodic	$-99.7\% \pm 0.01\%$	$-87.3\% \pm 5.1\%$
Smooth	$-89.8\% \pm 2.3\%$	$-88.2\% \pm 3.7\%$
Diagonal	$-80.3\% \pm 4.5\%$	$-84.6\% \pm 4.6\%$

Table 4.3: Percentage difference in MMD distance between the training and the test distribution after using SF and PSF.
Negative values denote a decrease in the distance between the marginal distributions. -100% indicates a reduction of the MMD distance of two orders of magnitude after applying SF or PSF.

	SF+SVM	***PSF+SVM***
Radial	$0.342 \to 0.779 \pm 0.03$	$0.342 \to 0.479 \pm 0.05$
Periodic	$0.488 \to 0.5 \pm 0.01$	$0.488 \to 0.568 \pm 0.02$
Smooth	$0.894 \to 0.476 \pm 0.06$	$0.894 \to 0.522 \pm 0.08$
Diagonal	$0.866 \to 0.584 \pm 0.02$	$0.866 \to 0.609 \pm 0.09$

	IW+LSPC	***SSA+SVM***	***DAE+SVM***
Radial	$0.336 \to 0.598 \pm 0.06$	$0.342 \to 0.342$	$0.342 \to 0.387 \pm 0.05$
Periodic	$0.488 \to 0.512 \pm 0.0$	$0.488 \to 0.488$	$0.488 \to 0.512 \pm 0.0$
Smooth	$0.865 \to 0.936 \pm 0.02$	$0.89 \to 0.89$	$0.894 \to 0.613 \pm 0.02$
Diagonal	$0.768 \to 0.786 \pm 0.01$	$0.86 \to 0.932$	$0.866 \to 0.514 \pm 0.0$

Table 4.4: Accuracy change when using different CSA systems on the four synthetic data sets. The table reports the raw accuracy before CSA to left of the arrow, and the accuracy after CSA to the right of the arrow. The performance for the SSA+SVM system reports a single figure because of the deterministic nature of the algorithm; for the others algorithms involving a random initialization, mean and standard error are estimated out of 10 independent trials.

	SF+SVM	***PSF+SVM***	***IW+LSPC***	***SSA+SVM***	***DAE+SVM***
Radial	$+\mathbf{127.8}\% \pm \mathbf{9.4}\%$	$+40.1\% \pm 14.7\%$	$+7.8 \pm 2.1\%$	0%	$+13.16 \pm 14.1\%$
Periodic	$+0.33\% \pm 0.7\%$	$+\mathbf{16.35}\% \pm \mathbf{5.0}\%$	$+4.91\% \pm 0\%$	0%	$+4.92 \pm 0.0\%$
Smooth	$-46.76\% \pm 7.0\%$	$-41.61\% \pm 9.0\%$	$+\mathbf{8.36}\% \pm \mathbf{1.3}\%$	0%	$-31.45 \pm 2.6\%$
Diagonal	$-32.54\% \pm 2.4\%$	$-29.63\% \pm 10.0\%$	$+2.74\% \pm 0.7\%$	$+\mathbf{8}\%$	$-40.65 \pm 0.0\%$

Table 4.5: Percentage change in accuracy when using different CSA systems on the four synthetic data sets.
The performance for the SSA+SVM system reports a single figure because of the deterministic nature of the algorithm; for the others algorithms involving a random initialization, mean and standard error are estimated out of 10 independent trials.

In contrast, PSF yields a significant improvement even when applied to the *radial* data set, due to the fact that it is able to extrapolate a periodic structure under the data; however, it no longer works on the *smooth* and the *diagonal* data sets because it cannot reconstruct a periodic structure from samples coming from a single period. In general, PSF continues to exhibit a high variance, reminding us that, despite the supervised guidance, the algorithm is very sensitive to its initialization.

Compared to SF and PSF, IW and SSA are more conservative, providing the best results on the data sets meeting their assumptions, and guaranteeing small or null improvements on the other data sets, but no big drops.

Finally, DAE achieves an improvement on the first two data sets (*radial* and *periodic*), but the accuracy is sensibly compromised when applied to the other two data sets (*smooth* and *diagonal*). Unfortunately, this behaviour is harder to explain in terms of theoretical assumptions, as they cannot be clearly expressed in explicit terms. The low performance may be due to the difficulty of discovering a low-dimensional manifold (Vincent et al., 2010) that preserves the structure of the conditional distribution of the labels.

Interestingly, beyond confirming the relationship between good classification performance and the satisfaction of all the conditions for successful CSA, this comparative study seems also to suggest that there may be a trade-off between the percentage change in performance and the strictness of the assumptions. The stricter the assumptions, the higher the percentage change, positive if the conditions are met or negative if the conditions are not met, as exemplified by SF in Table 4.5. On the other hand, if the assumptions are looser, the variation in performance is limited between the case when the conditions are met and the case when the conditions are not met, as illustrated by IW in Table 4.5. This phenomenon may be explained in the terms of the no-free lunch theorem (Wolpert and Macready, 1997).

4.5.3 Covariate shift adaptation via sparse filtering and periodic sparse filtering on real-world data sets

In this section, we validate the results obtained so far on real-world data sets. The aim of these experiments are: (i) proving that a classification system using PSF can provide a statistically significant improvement over the baseline system without CSA in a realistic scenario, and (ii) showing that PSF can provide a competitive performance against other CSA algorithms not only on crafted synthetic data but on real-world data as well.

Notice that when we worked with synthetic data in Section 4.5.2, we designed data sets whose structure would perfectly fit the assumptions of the CSA algorithms we considered. However, this ideal situation very rarely occurs when we deal with real-world data sets. Real-world data are very complex and they do not perfectly fit the simple assumptions of any of the CSA algorithms we examined. The assumptions of radial structure, periodicity, smoothness and PCA projectability can be satisfied at most to a limited degree. Also, notice that the high-dimensionality of the data and the impossibility to assess with precision its structure inevitably hinder our ability to provide a detailed explanation of the results as we did in the case of

	#Sp	*Language*	*Recording*	*Labelling*	*#Samp*	*#Pos Val*	*#Neg Val*
EMODB	10	German	acted	discrete (7 classes)	1211	289	922
DES	4	Danish	acted	discrete (5 classes)	974	579	395
VAM	47	German	natural	continuous (3 dimensions)	2495	167	2328
eNT	43	English	induced	discrete (6 classes)	2988	877	2111

Table 4.6: Comparison of ESR data sets.
#Sp refers to the number of speakers; *#Samp* refers to the number of 1-seconds samples; *#Pos Val* and *#Neg Val* refer respectively to the number of samples with positive valence and negative valence.

synthetic data. In interpreting our results we are limited to the use of indirect measures, such as classification accuracy.

For these simulations we decided to use emotional speech recognition (ESR) data sets for the following reasons. (i) It is well known to the affective computing community that these data sets are affected by covariate shift (Schuller et al., 2010). (ii) ESR data sets may, at least to some degree, comply with the assumption of PSF, in that they lend themselves to be modelled as user-dependent data, where each user could be specified by a different pdf $p\left(X^{(i)}\right)$ on a specific sub-domain $\mathcal{X}^{(i)}$, while the conditional distribution of the emotional labels may be approximately the same for all the users $p\left(Y|X\right)$.

Data sets. In the following experiments four well-known ESR data sets are employed: the Berlin Emotional Database (EMODB) (Burkhardt et al., 2005), the Danish Emotional Speech Database (DES) (Engberg and Hansen, 2007), Vera am Mittag (VAM) (Grimm et al., 2008) and eNTERFACE (eNT) (Martin et al., 2006). This collection of data sets is very heterogeneous, containing recordings from different speakers, in different languages, with different labels and collected with different protocols (see Table 4.6 for a schematic comparison of the data sets and Appendix B for a more detailed description of each data set). All the data sets are pre-processed as explained in Appendix B. Moreover, following the standards of the literature in CSA for emotional speech recognition, the samples are normalized per speaker (as discussed in Schuller et al., 2010) and the training and test data are upsampled in order to balance the classes using a simple re-sampling with repetition procedure (as suggested by Zhang et al., 2013). Labels are aligned as explained in Appendix B along the valence dimension; we chose to consider the valence labels (instead of arousal labels or emotional content labels) because they constitute a more challenging problem (Wollmer et al., 2008; Busso et al., 2007) on which to test our algorithms.

Experimental protocol. When performing CSA, we decided to use only speakers from EMODB, DES and eNTERFACE for testing, excluding VAM because of the high unbalance between the valence classes (see Table 4.6).

Data are then partitioned using the following protocol: in each trial, one speaker from either EMODB, DES or eNTERFACE is randomly selected; half of the samples from the selected speaker constitute the target set $\mathbf{X}^{\mathbf{tgt}}$, while the remaining half constitutes the test set $\mathbf{X}^{\mathbf{tst}}$; all the samples from the three remaining data sets constitute the training set $\mathbf{X}^{\mathbf{tr}}$. Adaptation is performed using the training and target data; classification uses training data for learning,

target data for model selection and test data for evaluation. This protocol has two advantages: (i) in line with the most challenging scenarios in the literature, this is a dataset-out and speaker-out scenario, in which the training set does not contain samples from the same data set or the same speaker in the test set, and (ii) preserving part of the samples only for testing allows us to properly evaluate the degree of inductive generalization.

To perform classification, we used the same systems implemented for the experiments on synthetic data in Section 4.5.2.

When performing adaptation with SF and PSF, we rely on the use of an early stopping criterion based on the estimation of the distance $D\left[\mathbf{Z^{tr}}, \mathbf{Z^{tgt}}\right]$ using the KS test. Following Hassan et al. (2013), we apply the KS test to derive a gross estimation of the distance between the distribution of the training $\mathbf{Z^{tr}}$ and target data $\mathbf{Z^{tgt}}$. We apply the KS test feature-by-feature $KS\left(Z^{tr}_{\cdot,j}, Z^{tgt}_{\cdot,j}\right)$, and we average over all the features $E\left[KS\left(Z^{tr}_{\cdot,j}, Z^{tgt}_{\cdot,j}\right)\right]$. We stop training at the point in which the learned distribution of the training data and target data achieves a minimum in the average KS distance over the first 50 iterations. This simple policy allows us to decrease the number of hyper-parameters, reduce the computational time and improve the results.

In order to evaluate the performance in this unbalanced classification problem we employ the unweighed average recall (UAR): $\frac{1}{C}\sum_{c=1}^{C} recall\,(c)$ (Batliner et al., 2010), where $recall\,(c)$ denotes the recall for class c. The UAR index gives a better estimation of the performance of a classifier in an unbalanced scenario by penalizing those classifiers that neglect a minority class by always predicting the majority class.

For each configuration, we report the mean and the standard error achieved over 10 independent simulations. In order to validate the contribution of PSF to classification, we perform a paired Wilcoxon test with the null hypothesis that the classification performance with and without CSA has the same mean (under the assumption that the results are symmetrically distributed around the true mean performance). Statistics for the hypothesis test are collected from 100 independent trials.

Results. Figure 4.6 shows the UAR of the different classification systems on the three data sets following the protocol described above.

After running the Wilcoxon test, we felt confident rejecting the null hypothesis that classification with and without PSF is equivalent in the case of EMODB and eNTERFACE (p-value for the null hypothesis, respectively, $9.5\cdot10^{-5}$ and $6.9\cdot10^{-4}$), but not in the case of DES (which could not be rejected at the significance level of 0.05). Thus, the statistical test implies that PSF was able to capture some relevant periodic structure in the conditional distribution when applied to the EMODB and eNTERFACE data sets, while the negative result on DES suggests that the better performance may have been due a random effect.

In general, the experimental results suggest that PSF is able to provide a performance better or comparable to those of the other CSA algorithms, with the exception of DAE which

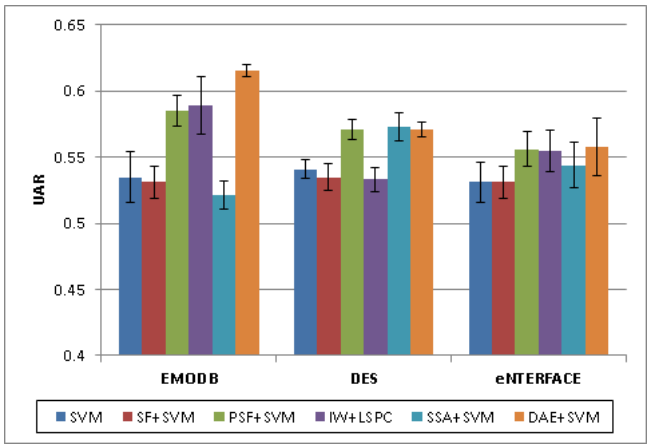

Figure 4.6: UAR accuracies of the different CSA classification systems along with the baselines.

outperformed PSF on EMODB. On the other hand, the SF algorithm failed to improve at all over the baseline. This is not surprising, and it can be easily explained by the fact that these real-world data sets are too complex to comply with the tight assumption of a radial data structure. Among the other CSA algorithms, IW yields the second best UAR on EMODB and a slightly worse UAR than PSF on eNTERFACE, but it failed to provide an improvement on DES, potentially because DES speaker data may lie in sub-domains removed from the other data sets. SSA performed the best on DES, yielded a UAR higher than the baseline (but lower than PSF, IW or DAE) on eNTERFACE, but performed the worst on EMODB; this could hint at the fact that the PCA components of the EMODB test speaker data are not easily projected on the PCA components of all the other speakers. Finally, DAE provided the best results with the highest UAR on EMODB and comparable UAR to other classification systems on the remaining data sets.

Interestingly, even when negative, the results of the CSA algorithms on individual data sets may allow us to get an insight into the structure of the conditional distribution; indeed, the failure to perform CSA may be explained by the fact that a specific data set does not meet the assumptions of the CSA algorithm. For instance, on EMOBD, the SSA algorithm fails to provide a significant improvement, hinting at the fact that the PCA space of the training and test data set cannot be linearly projected onto each other. Similarly, on DES, the failure of IW could point to the fact that the pdfs describing the DES speakers lie on different sub-domains or that they may have a discontinuous irregular conditional distribution. However, it is important to remember that the failure of a CSA algorithm may be due to other factors, and that this explanation in terms of unmet assumptions is a likely hypothesis but requires further validation.

These experiments illustrate and confirm the usefulness of properly devised FDL algorithms, such as PSF, for carrying out CSA. In the next section, we will conclude by summing up the results of this chapter.

4.6 Summary of the Chapter

In this section, we review the results achieved in this chapter, underlining their meaning and relevance.

Definition of the requirements for CSA. Our study set out with a clear definition of the necessary requirements to perform successful CSA. Such an understanding allowed us to consider the problem of CSA in a consistent way: instead of thinking of a CSA algorithm simply as a process reducing the distance between marginal distributions, we formalized it as a process reducing such a distance within the constraints of preserving relevant information for further supervised learning and classification. We expressed this understanding in two conditions that we used to guide our study of SF and PSF.

SF for CSA. We have showed theoretically that the SF algorithm is able to implicitly perform CSA under a precise assumption about the structure of the data. This result is intimately connected to the dynamics of SF and derives directly from the results in Chapter 3. The requirement for CSA translates into the conditional distribution of the data having a structure explained by cosine neighbourhoodness. If this condition is met, our study supports the thesis that SF is a suitable algorithm to perform CSA.

PSF. Motivated by our formal analysis of SF and our discovery of its limits, we have proposed the novel PSF algorithm that is able to perform CSA under less restrictive assumptions. We designed the new SF-like algorithm by following the guidelines we exposed in Section 3.5.1: we preserved the dynamics of sparsification to guarantee the infomax principle, and we tailored the informativeness principle around the requirement of preserving periodic structures. This allowed us to propose a new flexible and effective SF-like algorithm with a computational complexity equivalent to the original.

Supervision in PSF. PSF has been designed to be a more versatile algorithm than SF, one able to deal with more complex and realistic data structures. However, this increase in the learning capacity of PSF came at the cost of making learning more challenging; specifically, it became hard to direct the unsupervised algorithm to learn the desired periodic structure that would be relevant for further supervised learning. We therefore enriched the algorithm by making it able to rely on labelled information in order to discover and preserve relevant structures.

PSF for CSA. Through a theoretical analysis analogous to the one carried out for SF we were able to provide a firm ground for understanding the behaviour of PSF and for showing that it could indeed perform CSA under the requirement that the conditional distribution of the labels has a periodic structure.

Dependence of the success of CSA on the data structure. Our comparative experimental study used carefully-designed data sets to highlight that the success of a CSA algorithm

is strongly correlated with the satisfaction of necessary conditions for CSA, which, in turn, are often tied to the structure underlying the data being processed. We were then able to show that the success not only of SF and PSF, but of other classic algorithms like IW and SSA too, is dependent on the structure of the data.

SF and PSF in real-world scenarios. Experimental results clearly supported our formal analysis of SF and PSF, demonstrating strengths and weaknesses of the FDL algorithms both on synthetic and real-world data. These simulations proved that PSF can achieve good performance in line with other classic CSA algorithms. This feature, combined with its light computational cost, makes PSF a competitive candidate for CSA in real-world applications, whenever its assumptions are met.

SF and PSF for data structure exploration. By making the conditions required for CSA explicit, our analysis also allowed us to obtain meaningful results even in the case where a CSA algorithm fails. Indeed, in such an eventuality, it would be possible to gain insights about the structure of the conditional distribution of a data set by analysing the hypothesis that the failure is due to the violation of the assumptions. In our experiments, we could easily suggest similar hypotheses in the case of SF, PSF, IW or SSA, but not in the case of DAE, for which such conditions are not explicit. These hypotheses, however, are not certain conclusions, and they must be treated only as a sensible starting point for a deeper analysis of the data.

Sensitivity of PSF to initialization. While experimental results on PSF empirically verified our theoretical statements, it has been observed that the algorithm is quite sensitive to initialization, as the high variance among multiple trials with different initializations suggests. This hints at the fact that some instantiations led to very good solutions, while others learned less useful filters. We implemented a simple criterion based on the KS test to select those trials where the distance between the training and test data is minimized, but more refined criteria may be designed to provide better solutions. In addition, an alternative model selection procedure may be implemented to explore more thoroughly the space of the hyper-parameters; better values for the dimensionality of the learned space and for the value of the scaling parameters may be found, thus improving the final performance of PSF.

FDL for CSA. The conclusions about SF and PSF and their ability to perform CSA may be extended to the class of FDL algorithms. Even if the insensitivity of FDL with respect to the original pdfs $p(X^{tr})$ and $p(X^{tst})$ is only partially accurate, as FDL algorithms are sensitive to the structure of the data, our work suggests that FDL has the potential to be an effective and efficient framework for CSA. The theoretical framework we offered on how SF-like algorithms perform CSA could be directly exploited for the development of novel algorithms, which preserve the computational simplicity of FDL and, at the same time, define explicit assumptions on the conditional structure of specific real-world data sets. If prior knowledge is available about the conditional structure, the assumptions of new algorithms may be tailored to this knowledge; otherwise, metric learning algorithms may be employed or integrated in the CSA

algorithm in order to learn more about the structure underlying the data.

This concludes our study of SF and SF-like algorithms for CSA. In the next chapter we will discuss in more detail all the results obtained so far.

Chapter 5

Discussion

This chapter draws together the results we presented in the previous two chapters and discusses their relevance and implications. Moreover, taking these results as a starting point for future work, it suggests possible avenues for future research that follow from this dissertation and which may be further investigated.

Section 5.1 summarizes the work and the results we obtained, putting particular emphasis on their meaning and implications. Section 5.2 describes several potential avenues of research that may be pursued in the future.

5.1 Implications and Issues

Our study of FDL and its application to CSA was aimed at developing a clear and reliable understanding of the potentialities and the limitations of these algorithms in performing unsupervised learning. Tables 5.1 and 5.1 summarize the contributions provided in Chapters 3 and 4.

The study in Chapter 3 suggested a framework grounded on information-theoretic and optimization concepts in order to distinguish and analyse FDL algorithms. This approach provided a fruitful guideline for our ensuing analysis of the SF algorithm, allowing us to uncover

Contribution
✓ Definition of FDL in information-theoretic terms (see Section 3.1.3)
✓ Definition of SF in information-theoretic terms (see Section 3.3.9)
✓ Analysis of SF through representation filters (see Section 3.3.4)
✓ Interpretation of SF as a clustering algorithm (see Section 3.3.8)
✓ Uncovering of the strengths and limits of SF (see Section 3.3.2.5)
✓ Uncovering of ideal scenario for using SF (see Section 3.4.4)
✓ Definition of bounds for Euclidean distance for SF (see Section 3.3.7)
✓ Analysis of real-world applicability of SF (see Section 3.4.5)
✓ Insights on the development of novel FDL algorithms (see Sections 3.5.1.1 and 3.5.1.4)
✓ Analysis of new SF-like algorithms (see Sections 3.5.1.2 and 3.5.1.3)
✓ Interpretation of RP algorithms as FDL algorithms (see Section 3.5.2)

Table 5.1: Summary of the contributions of Chapter 3.

Contribution
✓ Definition of the requirements for CSA (see Section 4.1.2)
✓ Analysis of the conditions for using SF to perform CSA (see Sections 4.2.1 and 4.2.2)
✓ Definition of the novel PSF algorithm (see Section 4.3)
✓ Introduction of supervision in PSF (see Section 4.3)
✓ Analysis of the conditions for using PSF to perform CSA (see Sections 4.4.1 and 4.4.2)
✓ Uncovering the dependence of the success of CSA on the data structure (see Section 4.5.2)
✓ Application of SF and PSF to real-world scenarios (see Section 4.5.3)
✓ Use of SF and PSF for data structure exploration (see Section 4.5.3)
✓ Uncovering the sensitivity of PSF to its initialization (see Section 4.5)
✓ Insights on the development of FDL algorithms for CSA (see Section 4.1)

Table 5.2: Summary of the contributions of Chapter 4.

the principles of preservation of structure that are one of the reasons behind the success of SF. Our conclusions were then validated on synthetic and real-world data, thus giving further support to our theoretical statements. This study has thus provided an approach to the study FDL algorithms and it has produced explanatory insights into the specific SF algorithm, which can be useful both for the development and the deployment of new algorithms.

Despite these results, significant and relevant problems remain to be investigated. The analysis of the dynamics of SF can hardly be considered complete. While the standard behaviour has been closely examined in our study, limit-case behaviour provided by particularly challenging scenarios or unusual initializations may be object of future research. We argued that the success of SF may be explained through the lens of the preservation of the structure of cosine neighbourhoodness of the conditional distribution of the labels; we showed that this principle indeed explains the success of SF on synthetic data and real-world data. However, it has still to be shown that this principle holds for all the data sets on which SF has been successfully applied. Such a study, which would require retrieving and inspecting several different data sets, could either provide further confirmation of our results or highlight the potential for further deeper analysis. The extension of the conclusion we have achieved about SF remains also an open question. Our results easily generalize to SF-like algorithms, like PSF, whose behaviour is analogous to SF. However, radically different FDL algorithms may behave in completely different ways than SF. In this case, our conclusion may not hold and it may be necessary to carry out a new theoretical analysis from scratch. Finally, the exploration of the class of FDL algorithms, which we briefly attempted in Section 3.5, is far from being exhausted. We considered the natural set of SF-like algorithms and we reviewed RP algorithms as a case study, but new and radically different algorithms may be proposed and studied.

After concluding our study on FDL and SF, we considered in Chapter 4 the extension of FDL to tackle the problem of covariate shift. In this case, too, our analysis was informed by a conceptual analysis of the conditions necessary for CSA. These generic conditions, that would hold for any RL algorithm, were used to formally evaluate the possibility of performing CSA via SF. Moreover, building on the results and the guidelines for designing new SF-like algorithms that we proposed in the previous chapter, we designed the new PSF algorithm as a more versatile FDL algorithm able to perform CSA. The design procedure we followed may be

taken to be emblematic for the development of new SF-like algorithms and the results of PSF on real-world data will hopefully exemplify the potential of FDL for performing CSA.

This work on FDL and CSA leaves several challenges still open. Our research and our experimental simulations consider and analyse two concrete instances of FDL algorithms, SF and PSF. These algorithms are shown to work when a data set exhibits certain particular structures underlying its conditional distribution. These requirements may be quite stringent, and they may make us doubt about the practical applicability of our algorithms to real-world scenarios. In order to exploit the potential of the FDL approach to unsupervised learning it would be useful to develop solutions that may be less dependent on the structure of the data or which may adapt to specific data sets. Our experimental simulations showed that PSF can successfully perform CSA, but more experiments may be required to establish correctly the limits of its potentialities. Differently from SF, PSF requires the definition of several hyper-parameters, whose number is bound to grow with the number of classes being considered; this may prevent its application to more complex problems, unless effective policies for setting these hyper-parameters are defined. Also, the high variance exhibited by PSF (and SF) in our experiments constitute an open challenge which should be addressed in order to guarantee a more stable and predictable behaviour of the algorithm.

5.2 Further Work

In this section we explore future directions for the work on FDL and CSA. We suggest potential themes and avenues of research that could extend the research done in this dissertation.

Section 5.2.1 focuses on new conceptual developments, meant either to establish connections between FDL and other sub-fields of machine learning or to provide new interpretations of FDL. Section 5.2.2 deals with potential theoretical research, aimed at analysing more carefully and more rigorously FDL in order to better understand its properties and its potentialities. Section 5.2.3 discusses more applied research directions, suggesting ways in which FDL, SF, PSF may be extended and enriched in order to improve results or to define innovative and more versatile algorithms working under the i.i.d. or the covariate shift assumption.

5.2.1 Conceptual developments

Here we consider possible ways to further the development of a deeper understanding or new interpretations of FDL.

FDL interpreted using a feature learning formalism. Interestingly, our study of SF as an unsupervised learning algorithm shares a similar methodology with the very recent work done by McNamara et al. (2016) on feature learning. In their modular theory of feature learning, they argue that an unsupervised learning algorithm has to meet the following four sufficient conditions to guarantee with high probability a reduction in the risk of a supervised learner: (i) $p(X)$ has a given structure; (ii) $p(X, Y)$ shares a structure with $p(X)$; (iii) the unsupervised learning algorithm exploits the structure in $p(X)$; (iv) the supervised learner exploits the structure in the learned representations. Our analysis on SF can be re-cast in this framework

to show that SF applied to radial data does indeed meet these sufficient conditions: (i) $p(X)$ has a structure explained by cosine neighbourhoodness; (ii) $p(Y|X)$ shares the same structure as $p(X)$; (iii) SF relies on the cosine distance; (iv) a supervised learner, such as SVM, can exploit the new Euclidean structure in the learned representations. This analysis could further confirm that, with high probability, SF working on radial data contributes to the reduction of the risk in standard supervised learners. A connection with this modular framework for feature learning may provide fruitful intersections between our work on FDL and the work of McNamara et al. (2016) and inspire a constructive dialogue between the two approaches.

FDL interpreted using the causal learning formalism. It may be possible to relate the study of FDL and DDL with causal learning (Pearl, 2009). In particular, an alternative interpretation to distinguish between DDL algorithms and FDL algorithms may be provided by the dichotomy between CAUSAL MODELS and *statistical models* in causal learning. DDL algorithms may be seen as types of statistical models, that is models where learning is determined by constraints deriving directly from the joint distribution of the observed variables; FDL algorithms may be seen as types of causal models, that is models where the learning constraints are not expressible in terms of the joint distribution of the observed variables (Pearl, 2009). Interestingly, it has been stated that behind any causal claim there must be a causal assumption that cannot be supported directly from the joint distribution of the observed variables (Pearl, 2009), but must be grounded instead on external principles. This statement holds true for FDL, which, in defining specific properties of $p(Z)$ to be learned, must justify the learning of these properties externally with no reference to the data; in the case of SF, for instance, sparsity was justified on the ground of computational efficiency and biological analogies. Exploring the topic of the relationship between causal and statistical models may help developing a deeper and more formal understanding of the families of DDL and FDL algorithms and their relationships.

FDL interpreted in relation to the no free lunch theorem. The *no free lunch theorem* (Wolpert and Macready, 1997) for optimization and machine learning constitutes a very powerful theoretical tool to interpret and explain the results provided by our research. This theorem states that there is no optimal machine learning algorithm in absolute terms, but, in different domains, different algorithms may perform better (Murphy, 2012).

Our theoretical analysis and practical simulations on SF clearly conform with this theorem. We did not argue that SF was able to provide state-of-the-art performance in absolute terms, but we showed that there are implicit assumptions and constraints that make SF better suited for certain scenarios instead of others. In particular, interpreting it as a soft clustering algorithm and comparing it against other representation learning algorithms based on an Euclidean metric, we concluded that SF is not a better algorithm than Euclidean-based clustering algorithms, but that there is a specific set of problems (in which $p(Y|X)$ is explained by the cosine metric) where the performance of SF is excellent, balanced by a set of problems (in which $p(Y|X)$ is explained by the Euclidean metric or other metrics) where its performance is less outstanding.

Similarly, our theoretical study and experimental validation of FDL for CSA agrees with the no free lunch theorem. Indeed our point was not to argue that SF and PSF can perform good CSA in absolute terms, but to show that there are specific conditions under which they

do perform good CSA. This conclusion is exemplified in our synthetic experiments comparing SF and PSF to other classical CSA algorithms: when their conditions on the structure of the conditional distribution are met (that is, $p(Y|X)$ has a radial or a periodic structure), then SF and PSF can successfully perform CSA; however, if these conditions are not met, while the conditions of other CSA algorithms are met, then SF and PSF naturally under-perform against the alternative CSA algorithms.

In general, the no free lunch theorem provides a powerful resource to study FDL algorithms and their application to other problems such as CSA. Indeed, the theorem instructs us about the necessity of clearly evaluating and defining the domain within which an algorithm may be successful, and this may provide a precious guideline for future research.

5.2.2 Theoretical developments

Beside furthering a conceptual understanding of FDL, research may be done to improve the formalization of FDL and uncover new mathematical or statistical properties of FDL and its application to CSA.

Statistical properties of learned representations. In Chapter 3 we studied SF accepting the i.i.d. assumption stating that the available data $\mathbf{X}$ were independent samples from the same distribution; in Chapter 4 we made the more challenging assumption of covariate shift, stating that the samples $\mathbf{X}^{\mathbf{tr}}$ and $\mathbf{X}^{\mathbf{tst}}$ are still independent but come from different distributions. However, no assumption has been made nor any formal study has been done about the learned representations $\mathbf{Z}$ or $\mathbf{Z}^{\mathbf{tr}}$ and $\mathbf{Z}^{\mathbf{tst}}$. A peculiarity of the SF and PSF algorithms is that, during their processing they compute new representations of the data taking into account all the samples being processed at the same time. In step A3 of SF and PSF, each individual sample is rescaled by the ℓ_2-normalization with respect to all other samples. This processing may have relevant consequences: on one side, it may make the algorithms sensitive to outliers, on the other side, it may affect the property of the independence of the samples in the matrix of learned representations $\mathbf{Z}$. This last effect may be particularly worthy of study. Determining the statistical properties of $\mathbf{Z}$ is important if we are going to use the learned representation for further statistical machine learning; for instance, if we were to feed the learned representations to a classifier making the assumption of i.i.d. data, the loss of the property of independence may affect the results. It would then be useful to know and to possibly quantify this loss of independence among the samples. From this point of view, the adoption of SF and PSF under covariate shift may be evaluated as a potential trade-off between the property of independence and identical distribution: SF or PSF may compensate for covariate shift and align training and test samples to a similar distribution, but in doing so they may lose the independence of the samples.

Even more interesting it may be the study of whether this hypothetical trade-off between independence and identical distribution occurring during CSA is an intrinsic effect of FDL for CSA. A rigorous theoretical study could allow us to properly assess these dynamics and evaluate their ensuing implications.

Random matrix theory. A more rigorous study of FDL could be grounded in random matrix theory, that is, the branch of mathematics studying the statistical properties of matrices made up of random variables. Random matrix theory has been used to study random projections (Saxe et al., 2011) and it has also been used to evaluate optimal stopping criteria for SF (Lederer and Guadarrama, 2014). As the learning in SF and PSF is strongly dependant on the randomly-initialized weight matrix, a better understanding of the properties of this matrix may lead to a deeper understanding of the dynamics of the algorithms and of potential improvements. As we have seen in our experiments, SF and PSF tend to exhibit very high variance; random matrix theory may give us a grounded way to initialize the weight matrix or to set the dimensionality of the learned representations so as to limit the variance of our results.

Information geometry. A deeper theoretical understanding of the dynamics of SF and PSF may be developed in connection with manifold learning and information geometry (Amari, 2016). Indeed, the property of preservation of structure could be more formally explained in the framework of differential geometry by modelling the data samples as points on a Riemannian manifold. Relevant data structures (that we presented in terms of Euclidean or cosine distance) may then be described in terms of Riemannian metric tensors, and preservation properties may be studied in terms of the preservation of these tensors. Manifold learning and information geometry may thus offer a more rigorous approach for characterizing the representation filters instantiated by SF or PSF, and they may help us understand what alternative representation filters we could design.

Topology of the representation filters. In developing new alternative SF-like algorithms, we suggested the possibility of devising algorithms instantiating representation filters designed to match specific data structures. A more rigorous study of which representation filters may be instantiated and which data structures they match may be done by relying on the formalism of topology (Munkres, 2000). Topology could allow us to decide whether and which novel representation filters are worth developing, by evaluating if existing filters may or may not match the data structures we are interested in. For instance, topology may justify the development of PSF on the ground that the conical filters instantiated by SF cannot be transformed through any continuous deformation in periodic filters; PSF filters thus allow for the preservation of a type of structure that filters instantiated by SF are precluded from preserving. Topology could instruct us on what types of filters and, consequently, which FDL algorithms we may want to implement.

5.2.3 Implementative developments

Finally we propose some concrete ideas for developing new FDL algorithms and improving existing SF-like algorithms. Differently from the observations made on conceptual or theoretical developments, these suggestions are mainly empirical and are grounded on intuition and experience.

Optimization of SF algorithms for processing new data. A very simple and immediate improvement of the SF algorithm to make it more versatile in dealing with real-world situations is to adapt it to on-line scenarios. The current implementation of SF requires new test data to be processed either alongside the training data or in batches. An interesting topic of research would be to investigate how the SF computation varies as a function of the number of samples processed and which statistical properties of the output can be guaranteed.

Also, even if the computational overhead of processing batches is limited, simple and more efficient solutions may be devised to avoid this computation. For instance, it is easy to conceive an alternative version of SF that would the normalization parameters during the training phase and rely on them when processing individual test samples as soon as they are received at test-time. Again, it could be interesting to analyse how this solution may affect the output.

Composition of non-linearities. In Section 3.5.1 we studied the possibility of developing new FDL algorithms by starting from SF and editing it; in particular, we suggested the idea of changing the non-linearity in step A2 in order to define an SF-like algorithm generating different representation filters. In Section 4.3 we actually followed this direction and defined the PSF algorithm by substituting the absolute-value non-linearity with a sinusoidal non-linearity. However, it may be argued that a way to generate more interesting dynamics would be to compose together several non-linearities. Even if the behaviour of the algorithm would be substantially the same (as the composition of multiple functions can always be substituted by a single function given by their composition), this strategy could allow us to define new filters in a simple and flexible way. There remains, however, the problem of deciding which non-linear functions to combine; this challenge may be addressed either experimentally or through a theoretical study, as suggested above.

Deep FDL. A conceptually similar approach would be to stack together multiple FDL modules into a deep architecture. For instance, instead of composing together several non-linearities in step A2 (as suggested above), it could be possible to stack together multiple SF modules. This idea of developing deep SF machines was already suggested in the original SF paper (Ngiam et al., 2011) and partially explored by other authors (Romaszko, 2013; Goodfellow et al., 2013), even though without much success. At this time, it may be possible to reconsider this option and try to exploit the framework provided by this work to evaluate whether stacking multiple SF modules could make sense and when it could work. The same concept of infomax and informativeness that we applied to a single SF module may be exploited to understand the results of the computation of several SF modules stacked together. This would give us reliable ground to decide whether the idea of stacking multiple SF modules is worth pursuing or not. If stacking SF modules were to prove effective, then SF could constitute a new alternative layer that may be used to develop deep neural networks. Analogously, a similar analysis may be made to evaluate the potential of stacking PSF modules or other FDL modules.

Combining FDL and DDL. In this work we studied FDL algorithms, considering them as a stand-alone class of algorithms, separated and often in contrast with DDL algorithms. We argued that the two approaches of DDL and FDL are, in a sense, diametrically opposed.

However, this does not mean that we could not consider possible combinations or a blending of them. As already suggested in Section 3.1.1, certain algorithms could already be seen as borderline cases between the two families of algorithms. It may then be reasonable to explore this area between DDL and FDL more carefully; indeed, the two approaches may be potentially mixed in order to exploit the strengths of one and compensate for the shortcomings of the other. The work done here on understanding FDL algorithms may prove to be a precious asset for exploring this possibility: a better grasp of what FDL algorithms are doing and how they work may provide the required awareness for combining FDL and DDL algorithms in a reasonable and fruitful way.

FDL algorithms imposing new properties. It may also be possible to define entirely new FDL algorithms aimed at preserving or maximizing different properties of the learned distribution. SF imposes sparsity, RP imposes low dimensionality, but new properties may be conceived and hard-coded into new FDL algorithms. To do so we have to ask ourselves: *what other properties we may want to impose on the learned representation or on the feature distribution?*

Analysing regularization constraints used in machine learning literature may be a good starting point, as these constitute formal properties of a representation that are encoded as auxiliary objectives. Within the range of different regularization properties available, we have to restrict our attention to those regularization constraints that are applied directly to the learned representation (such as sparsity) and not to the learned model (such as weight decay). Sparsity is one of the most common regularization constraints, and, while SF imposes ℓ_1-sparsity, we may consider imposing different types of sparsity (Hurley and Rickard, 2009). Information-theoretic properties have a strong theoretical backing but also a high computational cost; despite this, it may be possible to devise energy-based models in which the learning of the energy surface is driven by an explicit principle of entropy minimization (LeCun et al., 2006). Biological properties may also provide ideas and inspiration for the definition of interesting properties to implement novel FDL algorithms (Carandini and Heeger, 2012; Ganguli and Sompolinsky, 2012). Another useful property that we may want to impose may be disentanglement (Bengio et al., 2013), that is the separation of the different information components that make up the data. Differently from the other objectives that are mainly formal-syntactic, separating information component may require some understanding of semantics. In the case of disentanglement, unsupervised adaptation, like SF, may be ineffective, and some form of supervised adaptation, like PSF, must be devised. Disentanglement may be a particularly useful property, especially when processing complex signals from which we want to extract different types of information (Rifai et al., 2012; Desjardins et al., 2012).

Alternatively, we may try to think of alternative FDL algorithms explicitly in terms of structure preservation. SF preserves collinearity, RP preserves relative distances. In this case, we have to ask ourselves: *what other structural properties are worth preserving?*

Alternative local structures defined in terms of geodesic distance on a manifold may be an interesting structural property; they may be defined to suit specific tasks and challenges, however it may be non-trivial to design FDL algorithms that preserve them.

Other topological structures in the original space may be preserved, beyond cones and periodic tiles. For instance, concentric tiles, which are not reducible to either cones or periodic tiles, may be an interesting structure to be preserved. Again, however, it may be far from trivial to design a FDL algorithm tailored to preserve certain structures.

This survey of research directions is meant to illustrate only a sample of the possible developments that may stem from further research on FDL algorithms and their application to CSA. Such a series of research topics is hopefully indicative of the potential possibilities and the promise of the FDL approach to representation learning.

In the following chapter, we will draw our final conclusions about this research.

Chapter 6

Conclusion

This chapter summarizes the results and contributions provided by the dissertation.

Our work started from the analysis of the family of FDL algorithms, a recently-proposed and innovative approach to unsupervised representation learning. Despite its practical successes, little theoretical study had been undertaken concerning the properties and the potentiality of these algorithms. Our analysis sets out with the aim of formalizing the idea of FDL algorithms and shedding light on the actual inner workings of its most successful implementation, SF. Our study developed through three consecutive stages.

First of all, we offered a conceptual discussion on the definition of FDL algorithms. In substitution for the intuitive description of FDL algorithms frequently used, we proposed a definition based on concepts grounded in information theory and optimization theory. We explained FDL algorithms as algorithms that explicitly maximize informativeness through a proxy and implicitly satisfy an infomax principle through the definition of constraints.

This understanding constituted a solid ground for the successive stage of inquiry, in which we provided a deep theoretical analysis of the SF algorithm. Indeed, our study succeeded in showing that SF explicitly maximizes informativeness through the proxy of sparsity and implicitly satisfies the infomax principle through the constraint of structure preservation. These insights allowed us to uncover the dynamics of SF, to interpret it as a clustering algorithm and to evaluate alternative SF-like algorithms which were considered in the literature but never formally studied. The same interpretation used in our analysis of SF was applied to RP algorithms, by suggesting a reading through the lens of the infomax and informativeness principles. Finally, these two principles were exploited again to suggest guidelines that could be used to devise new FDL algorithms.

Finally, the theoretical results we achieved informed the last stage of our study, the empirical validation of our statements. A set of simulations, both on synthetic and real-world data sets, was carried out to illustrate all the properties of SF that we uncovered. In particular, our experiments confirmed the dependence between the success of SF and the presence of a conditional distribution explained by a metric of cosine neighbourhoodness.

To sum up, this line of research offered the following high-level contributions:

- It constructed a new way to understand FDL algorithms that can be useful both to direct the study of existing algorithms as well as to guide the development of new ones;
- It offered a detailed analysis of the SF algorithm, which uncovered its assumptions and highlighted its strengths and weaknesses, thus providing reliable framework for deciding when to adopt this algorithm;
- It showed the way in which alternative SF-like algorithms can be analysed and validated, and how the paradigm of FDL can be used to interpret already existing algorithms.

After having achieved a deeper and more subtle understanding of FDL and SF, our attention turned towards the possibility of using these algorithms to perform CSA. Our interest was sparked by previous assertions about the insensitivity of FDL algorithms to the problem of learning the distribution of the data; such an insensitivity would provide a simple and effective way to work around the problem of covariate shift. In tackling this problem, too, our approach followed three consecutive stages.

In the first conceptual stage, we rigorously outlined what we meant by CSA for a representation learning algorithm. This allowed us to express our aims in two conditions: a marginal condition, requiring the reduction of the distance between the marginal distributions of the training and test data, and a conditional condition, requiring the preservation of the identity of the conditional distribution of the training and test data.

Once again, this understanding informed our ensuing theoretical analysis of concrete FDL algorithms for CSA. We started with SF: we showed that this algorithm meets the marginal condition, but it satisfies the conditional condition only in the particular case in which the conditional distribution of the labels has a radial structure. To overcome this limitation, we designed a new, more versatile supervised algorithm named PSF by exploiting the design principles for new FDL algorithms that we had previously developed. Analogously to what we did with SF, we carried out a formal analysis of the CSA properties of this algorithm: like SF, we showed that PSF meets the marginal condition; moreover, we showed that PSF is able to satisfy the conditional condition in the generic case in which the conditional distribution of the labels has a periodic structure.

Last, we found confirmation for our theoretical results in a set of empirical simulations. These experiments showed the necessity for our FDL algorithms to meet both the marginal and conditional condition in order to successfully perform CSA. By aligning SF and PSF with other CSA algorithms from the machine learning literature, we provided further confirmation for the argument that the success of these CSA algorithms depends on the tacit assumptions they make about the structure of the data.

This line of study on the application of FDL to CSA provided the following substantial contributions:

- It defined a neat set of conditions for evaluating the potentialities of FDL algorithms in carrying out CSA;
- It made explicit the conditions under which SF can perform CSA, thus uncovering the strengths and the limitations of this algorithm;

- It proposed the novel PSF algorithm and demonstrated its ability to perform CSA under the less strict requirement of the data having a periodic structure.

Work on the study of FDL algorithms and their application to CSA is far from being exhausted. We believe that this dissertation can open the way for further research on conceptual, theoretical and practical levels. Of the many potential future avenues of research, further work is being done at the moment to connect our results with the framework of feature learning proposed by McNamara et al. (2016) and with information geometry (Amari, 2016), with the aim of providing more rigorous formalizations. Research is also being done on the stimulating topic of developing new FDL algorithms designed in new forms radically different from SF.

It is our hope that this work may promote the study and the development of FDL algorithms. These algorithms offer an alternative and intellectually stimulating approach to representation learning. The simplicity and practical success of SF, combined with the deeper understanding of its dynamics that we offered in this dissertation, may constitute a starting point for a wider and more aware adoption of SF as well as for the development of new FDL algorithms.

Moreover, we argued in favour of the adoption of the FDL paradigm to perform CSA and we explored this possibility. CSA is a central and challenging problem in the development of algorithms that can be successfully deployed in real-world settings and research on this topic is commanding increasing attention. Again, we hope that this study on the FDL paradigm may provide ideas and insights that could be exploited to develop new solutions to the problem of covariate shift.

References

Tameem Adel and Alexander Wong. A probabilistic covariate shift assumption for domain adaptation. In *AAAI*, pages 2476–2482, 2015.

Michal Aharon, Michael Elad, and Alfred Bruckstein. k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, 2006.

Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, volume 19, page 41. MIT; 1998, 2007.

Andrew Arnold, Ramesh Nallapati, and William W Cohen. A comparative study of methods for transductive transfer learning. In *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pages 77–82. IEEE, 2007.

Baktash Babadi and Haim Sompolinsky. Sparseness and expansion in sensory representations. *Neuron*, 83(5):1213–1226, 2014.

Keith Ball. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31: 1–58, 1997.

Anton Batliner, Stefan Steidl, Dino Seppi, and Björn Schuller. Segmenting into adequate units for automatic recognition of emotion-related episodes: a speech-based approach. *Advances in Human-Computer Interaction*, 2010:3, 2010.

Anthony J Bell and Terrence J Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision research*, 37(23):3327–3338, 1997.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: a review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8): 1798–1828, 2013.

Yoshua Bengio et al. Deep learning of representations for unsupervised and transfer learning. *ICML Unsupervised and Transfer Learning*, 27:17–36, 2012.

Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2007.

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics, 2006.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

Margaret M. Bradley. Measuring emotion: The self-assessment manikin and the semantic differential. *Journal of Behavior Therapy and Experimental Psychiatry*, 25:49–59, 1994.

Pavel Brazdil, Christophe Giraud Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

Richard P Brent and Paul Zimmermann. *Modern computer arithmetic*, volume 18. Cambridge University Press, 2010.

Neil D.B. Bruce, Shafin Rahman, and Diana Carrier. Sparse coding in early visual representation: from specific properties to general principles. *Neurocomputing*, 171:1085–1098, 2016.

Felix Burkhardt, Astrid Paeschke, Miriam Rolfes, Walter F. Sendlmeier, and Benjamin Weiss. A database of German emotional speech. In *INTERSPEECH*, volume 5, pages 1517–1520, 2005.

Carlos Busso, Sungbok Lee, and Shrikanth Narayanan. Using neutral speech models for emotional speech analysis. In *INTERSPEECH*, 2007.

Emmanuel J. Candes, Justin K. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

Matteo Carandini and David J. Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62, 2012.

Kevin M Carter, Raviv Raich, and AO Hero. Learning on statistical manifolds for clustering and visualization. In *Communication, Control, and Computing, Allerton Conference on*, 2007.

Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions, 2007.

Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. *Semi-Supervised Learning*. MIT Press, 2006.

Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

Nisha Chhabra and Richa Dutta. Low quality iris detection in smart phone using k-mean algorithm. 2016.

Donald G. Childers, David P. Skinner, and Robert C. Kemerait. The cepstrum: a guide to processing. volume 65, pages 1428–1443. IEEE, 1977.

Adam Coates and Andrew Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *International Conference in Machine Learning (ICML)*, pages 921–928, 2011.

Adam Coates, Andrew Y. Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 215–223, 2011.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Sanjoy Dasgupta. Experiments with random projection. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 143–151. Morgan Kaufmann Publishers Inc., 2000.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random structures and algorithms*, 22(1):60–65, 2003.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

Jun Deng, Zixing Zhang, Erik Marchi, and Bjorn Schuller. Sparse autoencoder-based feature transfer learning for speech emotion recognition. In *Proceedings of Affective Computing and Intelligent Interaction*, 2013.

Jun Deng, Rui Xia, Zixing Zhang, Yang Liu, and Bjorn Schuller. Introducing shared-hidden-layer autoencoders for transfer learning and their application in acoustic emotion recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 4818–4822. IEEE, 2014.

Misha Denil and Nando de Freitas. Recklessly approximate sparse coding. *arXiv preprint arXiv:1208.0959*, 2012.

Guillaume Desjardins, Aaron Courville, and Yoshua Bengio. Disentangling factors of variation via generative entangling. *arXiv preprint arXiv:1210.5474*, 2012.

Zhen Dong, Mingtao Pei, Yang He, Ting Liu, Yanmei Dong, and Yunde Jia. Vehicle type classification using unsupervised convolutional neural network. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 172–177. IEEE, 2014.

Zhen Dong, Yuwei Wu, Mingtao Pei, and Yunde Jia. Vehicle type classification using a semisupervised convolutional neural network. *Intelligent Transportation Systems, IEEE Transactions on*, 16(4):2247–2256, 2015.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, 2001.

Nader Ebrahimi, Esfandiar Maasoumi, and Ehsan S Soofi. Ordering univariate distributions by entropy and variance. *Journal of Econometrics*, 90(2):317–336, 1999.

Bradley Efron. Bayesians, frequentists, and scientists. *Journal of the American Statistical Association*, 100(469):1–5, 2005.

Paul Ekman. An argument for basic emotions. *Cognition and Emotion*, 6:169–200, 1992.

Michael Elad. *Sparse and redundant representation*. Springer, 2010.

Charles Elkan. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.

Kjersti Engan, Sven Ole Aase, and J Hakon Husoy. Method of optimal directions for frame design. In *Acoustics, Speech and Signal Processing (ICASSP), 1999 IEEE International Conference on*, volume 5, pages 2443–2446. IEEE, 1999.

Inger Samsø Engberg and Anya Varnich Hansen. Documentation of the Danish emotional speech database. Technical report, Aalborg University, 2007.

Florian Eyben, Martin Wollmer, Alex Graves, Bjorn Schuller, Ellen Douglas-Cowie, and Roddy Cowie. On-line emotion recognition in a 3-d activation-valence-time continuum using acoustic and linguistic cues. *Journal on Multimodal User Interfaces*, 3:7–19, 2010a.

Florian Eyben, Martin Wollmer, and Bjorn Schuller. openSMILE - the Munich versatile and fast open-source audio feature extractor. In *Proceedings of ACM Multimedia*, 2010b.

Giovanni Fasano and Alberto Franceschini. A multidimensional version of the Kolmogorov–Smirnov test. *Monthly Notices of the Royal Astronomical Society*, 225(1):155–170, 1987.

Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Computer Vision, 2013 IEEE International Conference on*, pages 2960–2967, 2013.

Mário AT Figueiredo and Robert D Nowak. An EM algorithm for wavelet-based image restoration. *Image Processing, IEEE Transactions on*, 12(8):906–916, 2003.

Evelyn Fix and Joseph L Hodges Jr. Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, DTIC Document, 1951.

Luciano Floridi. *The philosophy of information*. Oxford University Press, 2011.

Peter Földiák and Malcom P. Young. Sparse coding in the primate cortex. *Handbook of brain theory and neural networks*, 1:1064–1068, 1995.

Edward W Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

Roman Frigg and Charlotte Werndl. A guide for the perplexed. *Probabilities in physics*, page 115, 2011.

Surya Ganguli and Haim Sompolinsky. Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual review of neuroscience*, 35:485–508, 2012.

Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. In *European Symposium on Artificial Neural Networks (ESANN)*, 2016.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *International Conference in Machine Learning (ICML)*, pages 513–520, 2011.

Hanlin Goh, Nicolas Thome, Matthieu Cord, and Joo-Hwee Lim. Unsupervised and supervised visual codes with restricted Boltzmann machines. In *Computer Vision–ECCV 2012*, pages 298–311. Springer, 2012.

Hanlin Goh, Nicolas Thome, Matthieu Cord, and Joo-Hwee Lim. Learning deep hierarchical visual feature coding. *Neural Networks and Learning Systems, IEEE Transactions on*, 25 (12):2212–2225, 2014.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2066–2073. IEEE, 2012.

Ian J. Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ramaiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Zhang Chuang, and Yoshua Bengio. Challenges in representation learning: a report on three machine learning contests. In *Neural Information Processing*, pages 117–124. Springer, 2013.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, and Aaron Courville. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1):723–773, 2012.

Michael Grimm, Kristian Kroschel, and Shrikanth Narayanan. The Vera Am Mittag German audio-visual emotional speech database. In *Multimedia and Expo, 2008 IEEE International Conference on*, 2008.

Roger Grosse. Differential geometry for machine learning, 2014. URL https://metacademy.org/roadmaps/rgrosse/dgml.

Zhongyi Gu, Lin Zhang, Xiaoxu Liu, Hongyu Li, and Jianwei Lu. Learning quality-aware filters for no-reference image quality assessment. In *Multimedia and Expo, 2014 IEEE International Conference on*, pages 1–6. IEEE, 2014.

Thomas Hach and Tamara Seybold. Spatio-temporal denoising for depth map sequences. *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, 7(2):21–35, 2016.

Hirotaka Hachiya, Masashi Sugiyama, and Naonori Ueda. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, 80:93–101, 2012.

William Edward Hahn, Stephanie Lewkowitz, Daniel C. Lacombe Jr, and Elan Barenholtz. Deep learning human actions from video via sparse filtering and locally competitive algorithms. *Multimedia Tools and Applications*, pages 1–14, 2015.

Yoonchang Han and Kyogu Lee. Hierarchical approach to detect common mistakes of beginner flute players. In *ISMIR*, pages 77–82, 2014.

Yoonchang Han and Kyogu Lee. Detecting fingering of overblown flute sound using sparse feature learning. *EURASIP Journal on Audio, Speech, and Music Processing*, 2016(1):2, 2016.

Yoonchang Han, Subin Lee, Juhan Nam, and Kyogu Lee. Sparse feature learning for instrument identification: effects of sampling and pooling methods. *Journal of the Acoustical Society of America*, 139(5):2290–2298, 2016.

Asif Hassan, Robert Damper, and Mahesan Niranjan. On acoustic emotion recognition: compensating for covariate shift. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21(7):1458–1468, 2013.

James J Heckman. Sample selection bias as a specification error (with an application to the estimation of labor supply functions), 1977.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Jiayuan Huang, Alexander J Smola, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, et al. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, volume 19, page 601. MIT; 1998, 2007.

Niall Hurley and Scott Rickard. Comparing measures of sparsity. *Information Theory, IEEE Transactions on*, 55(10):4723–4741, 2009.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

Edward T Jaynes and Oscar Kempthorne. Confidence intervals vs Bayesian intervals. In *Foundations of probability theory, statistical inference, and statistical theories of science*, pages 175–257. Springer, 1976.

Edwin T. Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Jing Jiang. A literature survey on domain adaptation of statistical classifiers. Technical report, University of Illinois at Urbana-Champaign, 2008. URL http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/da_survey.pdf.

Jagat Narain Kapur. *Measures of information and their applications*. Wiley-Interscience, 1994.

Samuel Kaski. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, volume 1, pages 413–418. IEEE, 1998.

Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 180–191. VLDB Endowment, 2004.

Samuel Kim, Panayiotis G. Georgiou, Sungbok Lee, and Shrikanth Narayanan. Real-time emotion detection system using speech: Multi-modal fusion of different timescale features. In *Proceedings of Multimedia Signal Processing*, 2007.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Minjoon Kouh. Information maximization explains the sparseness of presynaptic neural response. *Neural computation*, 2017.

Andrei S. Kozlov and Timothy Q. Gentner. Central auditory neurons have composite receptive fields. *Proceedings of the National Academy of Sciences*, page 201506903, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1785–1792. IEEE, 2011.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *International Conference in Machine Learning (ICML)*, pages 473–480. ACM, 2007.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1:0, 2006.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Johannes Lederer and Sergio Guadarrama. Compute less to get more: using ORC to improve sparse filtering. *arXiv preprint arXiv:1409.4689*, 2014.

John A Lee and Michel Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

Yaguo Lei, Feng Jia, Jing Lin, Saibo Xing, and Steven Ding. An intelligent fault diagnosis method using unsupervised feature learning towards mechanical big data. *Industrial Electronics, IEEE Transactions on*, PP:1, 2015.

Xiao Li and Jeff A Bilmes. A Bayesian divergence prior for classiffier adaptation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 275–282, 2007.

Yujia Li, Kevin Swersky, and Richard Zemel. Learning unbiased features. *arXiv preprint arXiv:1412.5244*, 2014.

Henry W Lin and Max Tegmark. Why does deep and cheap learning work so well? *arXiv preprint arXiv:1608.08225*, 2016.

Ying Lin, Jianjun Sun, Chengqi Li, Yan Ma, Yujie Geng, and Yufeng Chen. Deep learning for intelligent substation device infrared fault image analysis. In *MATEC Web of Conferences*, volume 55. EDP Sciences, 2016.

Ralph Linsker. An application of the principle of maximum information preservation to linear systems. In *Advances in Neural Information Processing Systems*, pages 186–194, 1989.

Hongying Liu, Qiang Min, Chen Sun, Jin Zhao, Shuyuan Yang, Biao Hou, Jie Feng, and Licheng Jiao. Terrain classification with polarimetric SAR based on deep sparse filtering network. In *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International*, pages 64–67. IEEE, 2016a.

Zhao Liu, Yang He, Yi Xie, Hongyan Gu, Chao Liu, and Mingtao Pei. Pedestrian detection using deep channel features in monocular image sequences. In *International Conference on Neural Information Processing*, pages 608–615. Springer, 2016b.

Raul HC Lopes, ID Reid, and Peter R Hobson. The two-dimensional Kolmogorov-Smirnov test. 2007.

David J.C. MacKay. *Information theory, inference, and learning algorithms*, volume 7. Cambridge University Press, 2003.

Alireza Makhzani and Brendan Frey. k-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415, 1993.

Adam Marblestone, Greg Wayne, and Konrad Kording. Towards an integration of deep learning and neuroscience. *arXiv preprint arXiv:1606.03813*, 2016.

Anna Margolis. A literature review of domain adaptation with unlabeled data. Technical report, University of Washington, 2011.

Stacy Marsella, Jonathan Gratch, and Paolo Petta. *Computational Models of Emotion*, chapter Computational Models of Emotion, pages 21–47. Oxford University Press, 2010.

O. Martin, I. Kotsia, B. Macq, and I. Pitas. The eNTERFACE 05 audio-visual emotion database. In *Proceedings of IEEE Workshop on Multimedia Database Management*, 2006.

Daniel McNamara, Cheng Soon Ong, and Robert C. Williamson. A modular theory of feature learning. *arXiv preprint arXiv:1611.03125*, 2016.

Shuang Mei, Hua Yang, and Zhouping Yin. Unsupervised-learning-based feature-level fusion method for Mura defect recognition. *Semiconductor Manufacturing, IEEE Transactions on*, 2017.

Guido F. Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2924–2932, 2014.

James R Munkres. *Topology*. Prentice Hall, 2000.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference in Machine Learning (ICML)*, pages 807–814, 2010.

Deanna Needell and Joel A Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, Spain, 2011.

Jiquan Ngiam, Zhenghao Chen, Sonia A. Bhaskar, Pang W. Koh, and Andrew Y. Ng. Sparse filtering. In *Advances in Neural Information Processing Systems*, pages 1125–1133, 2011.

Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision research*, 37(23):3311–3325, 1997.

Genevieve B Orr and Klaus-Robert Müller. *Neural networks: tricks of the trade*. Springer, 2003.

Andrés Ortiz, Francisco J Martínez-Murcia, María J García-Tarifa, Francisco Lozano, Juan M Górriz, and Javier Ramírez. Automated diagnosis of Parkinsonian syndromes by deep sparse filtering-based features. In *Innovation in Medicine and Healthcare 2016*, pages 249–258. Springer, 2016.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11(3):387–434, 2005.

Giancarlo Pastor, Inmaculada Mora-Jiménez, Riku Jäntti, and Antonio J. Caamaño. Mathematics of sparsity and entropy: axioms, core functions and sparse recovery. *arXiv preprint arXiv:1501.05126*, 2015.

Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Signals, Systems and Computers, 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 40–44. IEEE, 1993.

JA Peacock. Two-dimensional goodness-of-fit testing in astronomy. *Monthly Notices of the Royal Astronomical Society*, 202(3):615–627, 1983.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Karl Pearson. LIII. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

Marcello Pelillo and Teresa Scantamburlo. How mature is the field of machine learning? In *AI* IA 2013: Advances in Artificial Intelligence*, pages 121–132. Springer, 2013.

Jose C. Principe. *Information theoretic learning: Rényi's entropy and kernel perspectives*. Springer, 2010.

Novi Quadrianto, James Petterson, and Alex J. Smola. Distribution matching for transduction. In *Advances in Neural Information Processing Systems*, pages 1500–1508, 2009.

Joaquin Quiñonero-Candela. *Dataset shift in Machine Learning (Neural Information Processing Series)*. MIT Press, 2009.

R Raghavendra and Christoph Busch. Learning deeply coupled autoencoders for smartphone based robust periocular verification. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 325–329. IEEE, 2016.

Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *International Conference in Machine Learning (ICML)*, 2007.

Kiran B. Raja, R. Raghavendra, Vinay Krishna Vemuri, and Christoph Busch. Smartphone based visible iris recognition using deep sparse filtering. *Pattern Recognition Letters*, 57: 33–42, 2015.

Kiran B Raja, R Raghavendra, and Christoph Busch. Color adaptive quantized patterns for presentation attack detection in ocular biometric systems. In *Proceedings of the 9th International Conference on Security of Information and Networks*, pages 9–15. ACM, 2016a.

Kıran B Raja, R Raghavendra, and Christoph Busch. Collaborative representation of deep sparse filtered features for robust verification of smartphone periocular images. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 330–334. IEEE, 2016b.

Marc'Aurelio Ranzato, Cristopher Poultney, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, 2006.

Marc'Aurelio Ranzato, Y-Lan Boureau, and Yann LeCun. Sparse feature learning for deep belief networks. In *Advances in Neural Information Processing Systems*, 2007.

Ajita Rattani, Reza Derakhshani, Sashi K Saripalle, and Vikas Gottemukkula. ICIP 2016 competition on mobile ocular biometric recognition. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 320–324. IEEE, 2016.

Salah Rifai, Yoshua Bengio, Aaron Courville, Pascal Vincent, and Mehdi Mirza. Disentangling factors of variation for facial expression recognition. In *Computer Vision–ECCV 2012*, pages 808–822. Springer, 2012.

Edmund T. Rolls and Alessandro Treves. The relative advantages of sparse versus distributed encoding for associative neuronal networks in the brain. *Network: Computation in Neural Systems*, 1(4):407–421, 1990.

Lukasz Romaszko. A deep learning approach with an ensemble-based neural network classifier for black box ICML 2013 contest. In *Workshop on Challenges in Representation Learning, ICML*, 2013.

Adriana Romero, Petia Radeva, and Carlo Gatta. No more meta-parameter tuning in unsupervised sparse feature learning. *arXiv preprint arXiv:1402.5766*, 2014.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

Ron Rubinstein, Alfred M Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. volume 98, pages 1045–1057. IEEE, 2010.

Shaun K. Ryman, Neil D.B. Bruce, and Michael S. Freund. Temporal responses of chemically diverse sensor arrays for machine olfaction using artificial intelligence. *Sensors and Actuators B: Chemical*, 2016.

Sandeepkumar Satpal and Sunita Sarawagi. Domain adaptation of conditional probability models via feature subsetting. In *Principles of Data Mining and Knowledge Discovery, European Conference on*, pages 224–235. Springer, 2007.

Andrew Saxe, Pang W Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *International Conference in Machine Learning (ICML)*, pages 1089–1096, 2011.

Bjorn Schuller, Anton Batliner, Dino Seppi, Stefan Steidl, Thurid Vogt, Johannes Wagner, Laurence Devillers, Laurence Vidrascu, Noam Amir, Loic Kessous, and Vered Aharonson. The relevance of feature type for the automatic classification of emotional user states: Low level descriptors and functionals. In *INTERSPEECH*, 2007.

Bjorn Schuller, Matthias Wimmer, Lorenz Mosenlechner, Christian Kern, Dejan Arsic, and Gerhard Rigoll. Brute-forcing hierarchical functionals for paralinguistics: A waste of feature space? In *Acoustics, Speech and Signal Processing (ICASSP), 2008 IEEE International Conference on*, 2008.

Bjorn Schuller, Bogdan Vlasenko, Florian Eyben, Martin Wollmer, Andre Stuhlsatz, Andreas Wendemuth, and Gerhard Rigoll. Cross-corpus acoustic emotion recognition: Variances and strategies. *Affective Computing, IEEE Transactions on*, 1:119–131, 2010.

Björn Schuller, Anton Batliner, Stefan Steidl, and Dino Seppi. Recognising realistic emotions and affect in speech: state of the art and lessons learnt from the first challenge. *Speech communication*, 53(9):1062–1087, 2011.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Dandan Si, Yuanyuan Hu, Zongliang Gan, Ziguan Cui, and Feng Liu. Edge directed single image super resolution through the learning based gradient regression estimation. In *Image and Graphics*, pages 226–239. Springer, 2015.

Patrice Y. Simard, Yann A. LeCun, John S. Denker, and Bernard Victorri. Transformation invariance in pattern recognition - tangent distance and tangent propagation. In *Neural Networks: Tricks of the Trade*, pages 239–274. Springer, 1998.

Qing Song, Guan-Ming Su, and Pamela C Cosman. Hardware-efficient debanding and visual enhancement filter for inverse tone mapped high dynamic range images and videos. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 3299–3303. IEEE, 2016.

Jost Tobias Springenberg and Martin Riedmiller. Learning temporal coherent features through life-time sparsity. In *Neural Information Processing*, pages 347–356. Springer, 2012.

Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: introduction to covariate shift adaptation*. MIT Press, 2012.

Masashi Sugiyama, Hirotaka Hachiya, Makoto Yamada, Jaak Simm, and Hyunha Nam. Least-squares probabilistic classifier: A computationally efficient alternative to kernel logistic regression. In *Proceedings of International Workshop on Statistical Machine Learning for Speech Processing (IWSML2012), Kyoto, Japan*, pages 1–10. Citeseer, 2012.

Hao Sun, Huanxin Zou, and Shilin Zhou. Accumulating pyramid spatial-spectral collaborative coding divergence for hyperspectral anomaly detection. In *Computer Vision in Remote Sensing, 2015 ISPRS International Conference on*. International Society for Optics and Photonics, 2016.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Tatiana Tommasi, Patricia Novi, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. *arXiv preprint arXiv:1505.01257*, 2015.

Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1521–1528. IEEE, 2011.

Alan Mathison Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265, 1937.

Peter D Turney. The management of context-sensitive features: A review of strategies. *arXiv preprint arXiv:0212.0212037*, 2002.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Brendan van Rooyen and Robert C Williamson. A theory of feature learning. *arXiv preprint arXiv:1504.00083*, 2015.

Vladimir Naumovich Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

Pascal Vincent and Yoshua Bengio. Kernel matching pursuit. *Machine learning*, 48(1):165–187, 2002.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference in Machine Learning (ICML)*, pages 1096–1103. ACM, 2008a.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference in Machine Learning (ICML)*, pages 1096–1103. ACM, 2008b.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010.

Thurid Vogt, Elisabeth André, and Johannes Wagner. *Affect and Emotion in HCI*, chapter Automatic recognition of emotions from speech: A review of the literature and reccomendations for practical realisation, pages 75–91. Springer, 2008.

G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean. Characterizing concept drift. *ArXiv e-prints*, November 2015.

Junfeng Wen, Chun-Nam Yu, and Russell Greiner. Robust learning under uncertain test distributions: Relating covariate shift to model misspecification. In *International Conference in Machine Learning (ICML)*, pages 631–639, 2014.

Benjamin Willmore and David J Tolhurst. Characterizing the sparseness of neural codes. *Network: Computation in Neural Systems*, 12(3):255–270, 2001.

Martin Wollmer, Florian Eyben, Stephan Reiter, Bjorn Schuller, Cate Cox, Ellen Douglas-Cowie, and Roddy Cowie. Abandoning emotion classes - toward continuous emotion recognition with modelling of long-range dependencies. In *INTERSPEECH*, 2008.

Martin Wollmer, Florian Eyben, Bjorn Schuller, Ellen Douglas-Cowie, and Roddy Cowie. Data-driven clustering in emotional space for affect recognition using discriminatively trained LSTM networks. In *INTERSPEECH*, 2009.

David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *Evolutionary Computation, IEEE Transactions on*, 1(1):67–82, 1997.

Dongrui Wu, Thomas D. Parsons, and Shrikanth S. Narayanan. Acoustic feature analysis in speech emotion primitives estimation. In *INTERSPEECH*, pages 785–788, 2010.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, pages 521–528. MIT; 1998, 2003.

Makoto Yamada, Leonid Sigal, and Michalis Raptis. No bias left behind: Covariate shift adaptation for discriminative 3D pose estimation. In *Computer Vision–ECCV 2012*, pages 674–687. Springer, 2012.

Chao Yan, Frans Coenen, and Bailing Zhang. Driving posture recognition by convolutional neural networks. *IET Computer Vision*, 2015.

Junfeng Yang and Yin Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM journal on scientific computing*, 33(1):250–278, 2011.

Zhao Yang, Lianwen Jin, Dapeng Tao, Shuye Zhang, and Xin Zhang. Single-layer unsupervised feature learning with ℓ_2 regularized sparse filtering. In *Signal and Information Processing (ChinaSIP), 2014 IEEE China Summit & International Conference on*, pages 475–479. IEEE, 2014.

Fabio Massimo Zennaro and Ke Chen. Towards understanding sparse filtering: A theoretical perspective. *arXiv preprint arXiv:1603.08831*, 2016a.

Fabio Massimo Zennaro and Ke Chen. On covariate shift adaptation via sparse filtering. *arXiv preprint arXiv:1607.06781*, 2016b.

Libao Zhang, Xu Liang, and Jie Chen. Saliency analysis and region-of-interest extraction for satellite images by biological sparse modeling. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 2757–2761. IEEE, 2016.

Shaohua Zhang, Hua Yang, and Zhouping Yin. Performance evaluation of typical unsupervised feature learning algorithms for visual object recognition. In *Intelligent Control and Automation (WCICA), 2014 11th World Congress on*, pages 5191–5196. IEEE, 2014.

Zheng Zhang, Yong Xu, Jian Yang, Xuelong Li, and David Zhang. A survey of sparse representation: algorithms and applications. *Access, IEEE*, 3:490–530, 2015.

Zixing Zhang, Ju Deng, and Bjorn Schuller. Co-training succeeds in computational paralinguistics. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013.

Chuan Zhao and Zhipeng Feng. Application of multi-domain sparse features for fault identification of planetary gearbox. *Measurement*, 2017.

Appendix A

Implementation Details

For the sake of reproducibility, this appendix provides details and links to the code used in our simulations in the experimental sections (Section 3.4 and Section 4.5). We relied on the following algorithms and implementations:

- *SF:* we used a Python implementation of the SF algorithm provided by Jan Metzen (available at: https://github.com/jmetzen/sparse-filtering) and based on the Matlab code originally released by Jiquan Ngiam (available at: https://github.com/jngiam/sparseFiltering).
- *PSF:* we used our own Python implementation of PSF (available at: https://github.com/FMZennaro/PSF).
- *SVM:* we used the Python implementation of the SVM algorithm provided in the *scikit* library (available at: http://scikit-learn.org/)
- *KNN:* we used the Python implementation of the KNN algorithm provided in the *scikit* library (available at: http://scikit-learn.org/)
- *k-means:* we used the Python implementation of the k-means algorithm provided in the *scikit* library (available at: http://scikit-learn.org/)
- *GMM:* we used the Python implementation of the GMM algorithm provided in the *scikit* library (available at: http://scikit-learn.org/)
- *KS test:* we used the Python implementation of the KS test provided in the *scipy* library (available at: https://www.scipy.org/)
- *MMD distance:* we used a Python implementation of the MMD algorithm provided by Vincent Van Asch (available at: http://www.clips.uantwerpen.be/~vincent/thesis-software/) and based on the Matlab code originally released by Arthur Gretton (available at: http://www.gatsby.ucl.ac.uk/~gretton/mmd/mmd.htm).
- *IW+LSPC:* for the integrated IW+LSPC system, we used the Matlab code provided by Hirotaka Hachiya (available at: http://www.ms.k.u-tokyo.ac.jp/software.html#IWLSPC).

- *SSA+SVM*: for the integrated SSA+SVM system, we used the Matlab implementation of SSA provided by Basura Fernando (available at http://users.cecs.anu.edu.au/~basura/DA_SA/) and the SVM implementation available in the *Matlab* environment.
- *DAE+SVM*: we used a Python implementation of the DAE algorithm provided by Rajarshee Mitra (available at: https://github.com/rajarsheem/libsdae).

Appendix B

Emotional Data Sets

For the sake of reproducibility, this appendix provides details on the real-world data sets that we used and the pre-processing that we implemented for the experiments (Section 3.4 and Section 4.5).

B.1 Data Sets

In this section we give a brief overview of the ESR data sets that we used in our experiments on real-world data in Section 3.4.5 and 4.5.3.

EMODB. The Berlin Emotional (EMODB) data set (Burkhardt et al., 2005) is an audio collection of emotional utterances assembled between 1997 and 1999 at the Technical University Berlin. It was originally conceived for studies of prosodic features, articulatory features and verification by re-synthesis The creators labelled the samples emotionally adopting a discrete theory of emotion with 7 basic emotions: anger, boredom, disgust, fear, joy, sadness and the neutral state. It contains recordings of 10 non-professional actors (5 female and 5 male) uttering emotionally-coloured sentences. Every actor was required to utter 10 sentences; each sentence was repeated for every single emotion considered in the study; thus, in total, the database contains 700 sentences (10 actors $\times$ 10 sentences $\times$ 7 emotions) plus 100 additional sentences as a backup. In order to induce specific emotions, actors were suggested to use a self-induction strategy based on recalling emotionally-charged events from the memory. The recordings were performed in an anechoic chamber; actors, standing in front of a microphone, were free to gesture, but they were asked to keep at a constant distance from the microphone in order to maximize the quality of the recordings.

DES. The Danish Emotional (DES) data set (Engberg and Hansen, 2007) is an audio emotional database built within the framework of the VAESS-project at the Aalborg University. The recordings were collected in order to develop synthetic voices expressing emotions. Emotional labelling is based on a discrete theory of emotion distinguishing 5 basic emotions: anger, happiness, sadness, surprise and the neutral state. The database contains recordings of 4 Danish radio theatre actors (2 female and 2 male). Every actor was required to utter single words,

sentences and passages; in total, the database contains 260 utterances (4 actors × 13 utterances × 5 emotions) plus 81 additional recordings. The recordings were performed in a acoustically-damped sound studio with the support of two operators; a high-quality microphone was used to record the utterances.

eNTERFACE. The eNTERFACE (eNT) database (Martin et al., 2006) is an audio-visual emotional database. It was created as a collection of audio, video and audio-visual samples in order to test emotion recognition algorithms. Labelling follows closely Ekman's discrete theory of emotion (Ekman, 1992) distinguishing six main emotional states (anger, disgust, fear, happiness, sadness and surprise) plus the neutral state. The emotions in the database are induced emotions: 42 subjects (8 female and 34 male) were given emotionally-charged stories to read in order to immerse themselves in specific situations; the subjects were then required to utter emotionally-coloured sentences. All the subjects were required to read and to express themselves in English, even if many of them came from backgrounds as diverse as Belgium, Cuba or Russia. In total the database contains 1166 samples. The recordings were performed in a small room using high-quality microphones and cameras and asking all the subjects to keep at a constant distance from the microphone.

VAM. The Vera Am Mittag (VAM) corpus (Grimm et al., 2008) is an audio-visual emotional database. It was created as a collection of natural emotion recordings in order to generate a high-quality audio-visual database for research in affective computing. VAM is made up of three databases: VAM-Audio, VAM-Video and VAM-Faces, containing, respectively, audio samples, video samples and face still images. The only data set we consider is VAM-Audio. Labelling is performed following a continuous theory of emotion with 3 dimensions: intensity, valence and dominance. Annotations were made using the self-assessment manikins method (Bradley, 1994): each annotator was required to rank intensity, valence and dominance of an emotion on a 5-values discrete scale; the annotations of all the annotators were then averaged to produce the final pseudo-continuous values of intensity, valence and dominance. All the samples in the data sets constitutes samples of natural emotions: the recordings are extracted from episodes of "Vera am Mittag" ("Vera at Midday"), a German talk show during which the guests discussed emotionally-charged topics. 47 guests (36 female and 11 male) were selected to be included in the corpus, as they showed a wide enough range of emotions and their recordings were of a quality good enough for audio processing. In total, the database contains 1018 recordings.

B.2 Preprocessing

In this section we provide details and justifications for the pre-processing that we performed on the emotional data we collected.

Segmentation. Segmentation of speech is a key problem when dealing with acoustic emotion recognition, especially when we are expecting to deploy a model in an on-line setting. Differently from other fields, there are no fixed agreed standards on what should be the ideal dimension

for emotional speech signal segments (Deng et al., 2013). Samples vary wildly in length from very short segments (like words or bursts) through entire utterances (like sentences or turns in a dialogue). Harmonizing, at least to some degree, the length of the samples is required in order to carry out a consistent analysis. Defining a common unit of segmentation means determining the atomic unit of analysis for emotional speech recognition: this unit should be long enough to contain emotional information related to emotions, but short enough to be treated as a stationary signal (Vogt et al., 2008). A simple and widespread approach consists in defining absolute time intervals and to segment recordings into frames of equal length. This segmentation procedure is fast and it leads to the creation of frames that, because of their uniform length, can be easily processed (Schuller et al., 2008). The segmentation length can vary from as little as tens of milliseconds (Eyben et al., 2010a) up to several seconds (Kim et al., 2007), even if intermediate values in the range of hundreds of milliseconds or seconds are more common (Schuller et al., 2007). Following the results available in the literature, and in force of preliminary experiments, we decided to segment all the recordings using a fixed one-second segmentation policy. Other methods for segmentation (such as relative segmentation, Schuller et al., 2008, or voice-activation-detection segmentation, Eyben et al., 2010a; Wollmer et al., 2009) are considered in the literature but they are beyond the scope of this work.

Feature extraction. All the samples are converted into standard feature representation vectors based on Mel-frequency cepstrum coefficient (MFCC) features (Eyben et al., 2010b) using the open-source platform OpenSMILE[1]. For each 1-second sample we compute features on 2 *domains* (raw, delta), extracting 12 *descriptors* (12 MFCC) and computing 3 *statistical operators* (mean, standard deviation and range), for a total of 72 features.

Label alignment. A standard policy for aligning different data sets in emotional speech recognition consists in defining binary emotional labels and then map discrete and continuous labels to the two binary categories. Standard binary classes are defined along the following dimensions: *emotional content*, discriminating between presence of emotion and the neutral state; *arousal* or *intensity* level, distinguishing between a state of high arousal with strong and intense feelings against a state of low arousal denoting mild and weak feelings; and, *valence* or *appraisal*, discriminating between between a state of positive valence characterized by a pleasant feeling against a state of negative valence characterized instead by an unpleasant feeling.

Table B.1 reports the mapping of different emotional classes to the emotional content binary categories; Table B.2 and Table B.3 respectively report the mapping to the arousal categories and to the valence categories following the standard policy defined in Schuller et al. (2010).

Table B.4 reports the number of samples contained in each data set, and the number of instances for each category we defined.

[1] http://audeering.com/technology/opensmile/

	Emotion	***No Emotion***
EMODB	anger, boredom, disgust, fear, joy, sadness	neutral
DES	anger, happiness, surprise, sadness	neutral
VAM	(all)	(none)
eNT	anger, disgust, fear, joy, sadness, surprise	(none)

Table B.1: Mapping of labels specific to each data set onto binary emotional content classes.

	High Arousal	***Low Arousal***
EMODB	anger, fear, joy	boredom, disgust, neutral, sadness
DES	anger, happiness, surprise	neutral, sadness
VAM	$arousal > 0$	$arousal < 0$
eNT	anger, fear, joy, surprise	disgust, sadness

Table B.2: Mapping of labels specific to each data set onto binary arousal classes.

	Positive Valence	***Negative Valence***
EMODB	joy, neutral	anger, boredom, disgust, fear, sadness
DES	happiness, neutral, surprise	anger, sadness
VAM	$valence > 0$	$valence < 0$
eNT	joy, surprise	anger, disgust, fear, sadness

Table B.3: Mapping of labels specific to each data set onto binary valence classes.

	#Orig Samples	***#1-sec Samples***	***#Emo***	***#Nemo***	***#HAro***	***#LAro***	***#PVal***	***#NVal***
EMODB	535	1211	1065	146	530	681	289	922
DES	260	974	778	196	564	410	579	395
VAM	947	2495	2495	0	1091	1404	167	2328
eNT	1287	2988	2988	0	1947	1041	877	2111

Table B.4: Number of samples in each data set and number of instances in each category. *#Orig Samples* denotes the number of original samples; *#1-sec Samples* denotes the number of samples after 1-second segmentation; *#Emo* denotes the number of samples with emotional content; *#Nemo* denotes the number of samples with neutral content; *#HAro* denotes the number of samples with high arousal; *#LAro* denotes the number of samples with low arousal; *#PVal* denotes the number of samples with positive valence; *#NVal* denotes the number of samples with negative valence.