UNIVERSITY OF PADOVA

DOCTORAL THESIS

Learning interpretable representations for classification, anomaly detection, human gesture and action recognition

Author:
Matteo TERZI

Supervisor:
Gian Antonio Susto

Co-supervisor:
Alessandro Beghi

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in

Brain, Mind and Computer Science
Department of General Psychology

Cycle XXXII

2019

UNIVERSITY OF PADOVA

Abstract

Brain, Mind and Computer Science
Department of General Psychology

Doctor of Philosophy

Learning interpretable representations for classification, anomaly detection, human gesture and action recognition

by Matteo TERZI

The goal of this thesis is to provide algorithms and models for classification, gesture recognition and anomaly detection with a partial focus on human activity. In applications where humans are involved, it is of paramount importance to provide robust and understandable algorithms and models. A way to accomplish this requirement is to use relatively simple and robust approaches, especially when devices are resource-constrained. The second approach, when a large amount of data is present, is to adopt complex algorithms and models and make them robust and interpretable from a human-like point of view. This motivates our thesis that is divided in two parts.

Sparse representations for Human Activity Recognition and Anomaly Detection

The first part of this thesis is devoted to the development of parsimonious algorithms for action/gesture recognition in human-centric applications such as sports and anomaly detection for artificial pancreas. The data sources employed for the validation of our approaches consist of a collection of time-series data coming from sensors, such as accelerometers or glycemic.

The main challenge in this context is to discard (i.e. being invariant to) many nuisance factors that make the recognition task difficult, especially where many different users are involved. Moreover, in some cases, data cannot be easily labelled, making supervised approaches not viable.

Thus, we present the mathematical tools and the background with a focus to the recognition problems and then we derive novel methods for:

- gesture/action recognition using sparse representations for a sport application;
- gesture/action recognition using a symbolic representations and its extension to the multivariate case;
- model-free and unsupervised anomaly detection for detecting faults on artificial pancreas.

These algorithms are well-suited to be deployed in resource constrained devices, such as wearables.

Robust and interpretable Deep Representations

In the second part, we investigate the feasibility of deep learning frameworks where human interpretation is crucial. Standard deep learning models are not robust and, unfortunately, literature approaches that ensure robustness are typically detrimental to accuracy in general. However, in general, real-world applications often require a minimum amount of accuracy to be employed. In view of this, after reviewing some results present in the recent literature, we formulate a new algorithm being able to semantically trade-off between accuracy and robustness, where a cost-sensitive classification problem is provided and a given threshold of accuracy is required. In addition, we provide a link between robustness to input perturbations and interpretability guided by a physical minimum energy principle: in fact, leveraging optimal transport tools, we show that robust training is connected to the optimal transport problem. Thanks to these theoretical insights we develop a new algorithm that provides robust, interpretable and more transferable representations.

Acknowledgements

This is the part I hate most. There are so many people that explicitly and implicitly contributed to this project and to my achievements, that the risk of forgetting to mention somebody is very high. I apologize in advance to those I have forgotten to mention.

I want to thank my advisor Gian Antonio Susto, co-supervisor Alessandro Beghi and Angelo Cenedese who gave me the chance to meet great researchers, to improve my scientific approach, and to guide me in my moments of indecision. I learnt a lot from them and I hope they have got something back from me.

I thank my many colleagues, Alberto, Enrica, Enrico, Chiara, Giulia, Daniele, Irene, Marco M., Marco F, Enrico P., Luca, Valentina and for sure many others that I do not remember in this moment. I really enjoyed all the work-days and social activities with them.

I thank professors, colleagues and friends who I met during my period abroad in Los Angeles. In particular, I thank professor Stefano Soatto for giving me the chance of visiting his lab, Antonio, Pratik and Alessandro, for the many moments together and the fruitful discussions, continued even after the period.

I thank my lifelong friends Umberto, Johnny, Carle, and more recent (but not least) ones, Alberto, Mattia, Nicola and Ortensia. We have known each other for a lifetime and I still don't understand how you guys can suffer me sometimes. We have had and will have so many great moments together.

And of course, my family. Apart from the material ones, there are many other reasons to say thank you. I am here writing and what I am also because of you.

And finally my companion in life Giorgia. We have been supporting and bearing each other every day for 14 years. I will never to stop to thank you for giving me the chance to improve my attitude, not always perfect, to feed my passions and for sharing the many moments we had and will have together.

Comments

Two brief comments are due. The first is about notation: the two parts of this thesis have (partial) different notations. This is because they describe different things and we wanted to be as adherent as possible to their respective common notation used in literature. The second comment is to justify the fact that in the second part, the used data sources are images and not time-series. The main motivations are:

- common benchmarks, results and public code are available nowadays for images, while there is no common and well-established benchmark for time-series classification;
- images are much easier to understand than time-series. Particularly, it is easy to see the effect of models on images;
- data types and architectures used for images and time-series are very similar. In fact, while the dimensionality is different both type of data can be considered as the discretization of continuous processes. This means that the methods presented in this manuscript can be easily extended to time series.

Contents

Part I

Sparse representations for Human Activity Recognition and Anomaly Detection

Chapter 1

Motivations

The objective of this short chapter is to review the main applications using wearable devices and time-series data and to present the main issues when dealing with such type of data for classification and anomaly detection tasks.

1.1 Applications

In the recent 10 years, remarkable advances in electronics led to the possibility of building smartphones and wearable devices equipped with miniaturized sensors such as GPS, Micro Electrical Mechanical Systems (MEMS) and body sensors. At the same time, research on Internet of Things (IoT) focused on developing protocols and architectures apt to handle an enormous number different data sources and types, that can archived and analyzed in cloud systems.

Fig. 1.1 and Fig. 1.2 represent common schemes for an IoT system for different applications: data is collected and processed in a decentralized fashion and then sent to the 'cloud' from which it is possible for other agents, such as a physician, to analyze the data or post-process them.

Thus, the general structure is formed by two entities: decentralized and centralized parts.
The decentralized part is the system directly interfaced with the user such a wearable device. This part performs pre-processing and very cheap computations. Usually, devices are usually resource-constrained. The centralized part receives and organizes data from all the decentralized systems generally in servers. The computational capability of this part is very high with respect to the decentralized one, which means that algorithms can be possibly memory and computational demanding in order to solve complex tasks, where huge amounts of data can be leveraged.

One of the fields thats has been witnessing a large attention in the scientific community is monitoring the human activity/gesture and its classification. Activity Recognition (AR) is a prominent research area with applications to home automation (Belgioioso et al., 2014), gaming (Gowing et al., 2014), sport (Cenedese, Susto, and Terzi, 2016) and health care (Clifton et al., 2013). In particular, the rapid growth of IMUs (Inertial-Measurement Units) has allowed, in recent years, the development of compact sensor-equipped devices (e.g. smartwatches and smartphones), which led efficient monitoring of human activities to be feasible and to have a strong impact on the quality of life (Clifton et al., 2013). In this context, related applications are performance analysis, especially for professional athletes but also common people. An example, is the so-called "Prius effect", a phenomenon coined when the hybrid Toyota Prius introduced real time feedback on gasoline consumption. It was observed and documented that a large subset of drivers would respond to the data by driving in a manner that decreased fuel consumption. Since the first two chapters of this part have a focus on human activity recognition, it is important in this context to first differentiate between *activities* and *gestures*. Gestures are in this work considered as basis movements that compose an activity that is conversely completely characterized by one or more atomic gestures: for example, in swimming, a stroke (gesture) completely characterizes the style (activity). Due to the vastness of application scenarios, it is helpful to categorize the AR problems into three main types:

- *continuous-repetitive* - activities that are continuous and composed by repeated gestures with a periodic behaviour within the same activity type;
- *continuous-spot* - continuous activities with non-repetitive gestures;
- *isolated* - activities composed by isolated gestures.

Health-care is probably the second field that most benefited from and that has been revolutionised by wearable data. Few examples are:

- Human parameter monitoring
- Diabetes – glucose monitoring
- Cardiopathy – ECG monitoring
- Epilepsy – attack prevention
- Neurodegenerative pathologies

In fact, in this thesis, we will also consider the problem of detecting faults on the pump of the so-called artificial pancreas (AP), a system that allows to monitor glucose and control it thorough the injection of insulin.

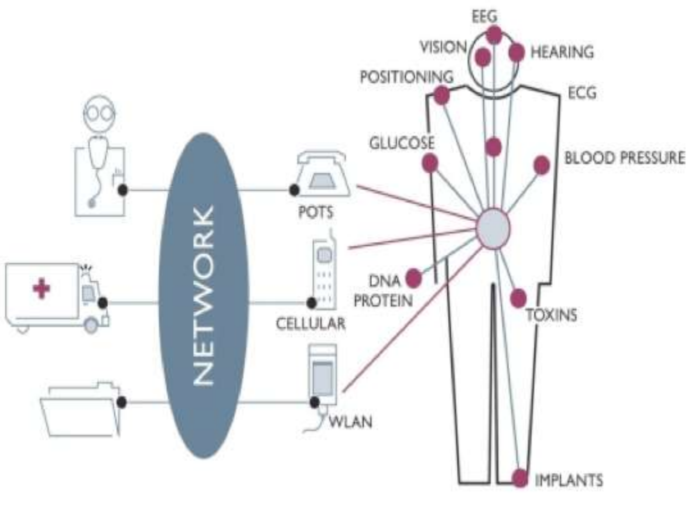


FIGURE 1.1

1.2 Issues and challenges

Challenges are of different nature: application-based and data-based. In AR but also in AP, the main application-based issue is due to the lack of resource of wearable devices in terms of computational capability and memory, which calls for the design

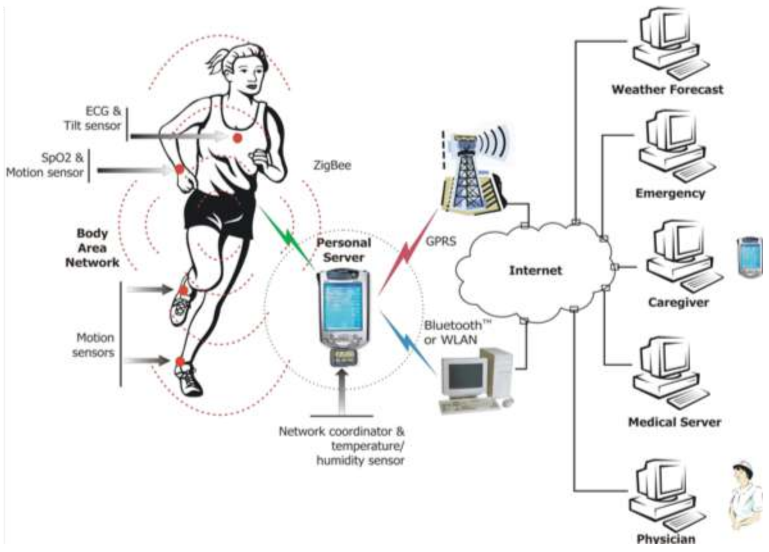


FIGURE 1.2

of algorithmic solutions that explicitly take into account these issues. Common solutions to this problem, is to reduce the informative context through *dimensionality* and *numerosity reduction*. We will review the most-known reduction techniques in Sec. 2.2.1 and apply them in Chapter 4.

The data-based issues are related to the data type, that is, time-series. Dealing with such kind of data presents many challenges:

- nuisance factors
- data heterogeneity
- time-correlation and streams of data

In the following we briefly discuss them and then in Chapter 2 we present the Machine Learning (ML) techniques presented in literature to (partially) address them.

1.2.1 Nuisance factors

By nuisance factors, we mean all the factors to which a task should be invariant. In the context of AR, we can list some of them:

- sensors' noise.
- time warping due to different acquisition frequencies
- orientation of the device

Noise. Devices sensors are cheap and often the signal is corrupted by white and coloured noise but also by a discrete amount outliers.
Time warping. The sampling frequency can change also in the same device. This causes a warping on the signal that impact of performance. For example, if we were to use a standard Euclidean distance to compare to signals, it would fail. Common solutions are interpolation-based; however, they only work when signal's delimiters are known (e.g. start and stop of a gesture).
Orientation. For wearable devices and applications based on MEMS signals, the change of orientation causes a change on the raw signal. In professional devices this

can be corrected using a navigation system, usually based on Kalman filters (Groves, 2013). However, in resource-constrained devices, it not possible to run complex systems. We would like to remark that in general, orientation can change dynamically. In the case of static change, in order to get features invariant to rotation, it is sufficient to apply the Singular Value Decomposition technique on the raw signal $X = U^T SV$ and discard U resulting in $\tilde{X} = SV$.

Thus, it is necessary to design algorithms that are robust to these factors without affecting accuracy and performance.

1.2.2 Data heterogeneity

The input data are highly variable due to user-dependency. In fact, there are intra and inter-differences. For example, different people perform gestures in a different way due to difference in height etc. However, it may also happen that two different people perform a gesture similarly while the same person performs it differently in different conditions. Thus, in user-tailored applications, invariant and fine-grained feature descriptors are required at the same time. The specialized literature reports different ML and pattern recognition techniques to classify a gesture from the captured data, from simple models such as k-Nearest Neighbour to more complex ones as Hidden Markov Models and Support Vector Machines (SVMs) (e.g. (Liu, Pan, and Li, 2010; Wu et al., 2009)). In such a context, the SVMs methods result to be the most effective but they may suffer from not being parsimonious.

1.2.3 Time-correlation and streams of data

Time series data are not i.i.d. (independent and identically distributed) and thus common ML approaches are not directly applicable. We will especially face this problem in Chapter 5. Moreover, many phenomena are distinguishable by how different time-point are correlated, that is by their dynamics. This is to say that correlation in this case is a feature we have to account for and not to discard. In addition, in general data manifest themself as a continuous stream and this poses the problem, particularly in AR tasks, of how to window the signal. A too large window may be expensive to analyse and lose peculiar features; on the contrary, a narrow windows may discards global distinctive features. Common strategy to tackle these problem are to consider time-series via the space of dynamical models and compare them directly in this space (see Sec. 2.3.2).

Chapter 2

Background on time-series classification

In this chapter, we will try to provide an overview of the main ML techniques that are used in the context of classification. Without aiming at being exhaustive, the main goal of this contribution is to highlight the differences among the approaches in terms of information they can provide and issues in their usage. In particular, we will use the term *classification* to indicate the sub-field of ML in the realm of *supervised learning* where "supervised" indicates that the output is known: given a signal $\mathbf{x}$ belonging to some domain $\mathbb{X}$ as an input and a finite set $\mathbb{Y}$ of different classes (the output), the problem of supervised learning consists on finding a rule that associates $\mathbf{x}$ to one $\mathbf{y} \in \mathbb{Y}$. Generally, the set of output classes (also called dictionary) is obtained according to a training procedure where a training input dataset is used to characterize both $\mathbb{Y}$ and the learning rule.

2.0.1 Notation

Throughout this chapter, we consider a time-series $\mathbf{z}_\bullet$ as a (finite-length) sequence of n ordered real-values at time-instants $t_{\bullet,1}, \ldots, t_{\bullet,n}$. For the sake of simplicity, and without loss of generality, we assume the time-series are obtained through a pre-processing phase that may include sampling and windowing of the continuous data flow coming from a measurement unit. The time-series is then characterized by p input descriptors $\mathbf{x}$ (whose meaning will be clearer in the following), hence the input space $\mathbb{X}$ is p-dimensional, and a training set composed by N signals $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ allows to define the class set $\mathbb{Y}$. These basic definitions and notation are depicted in Figure 2.1 and summarized in Table 2.1.

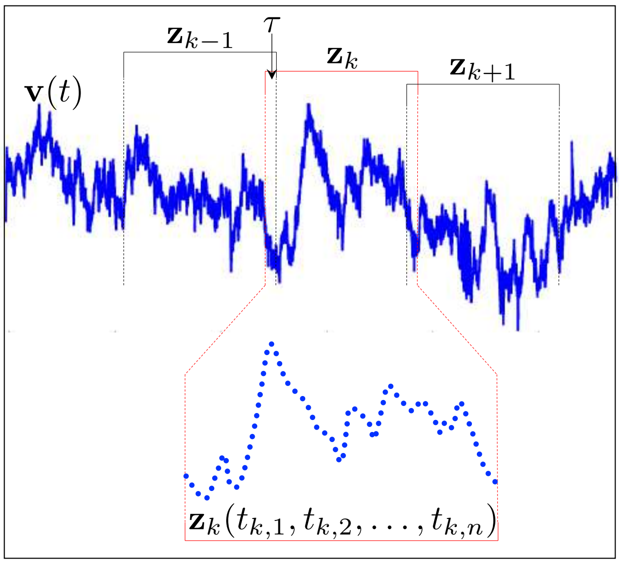


FIGURE 2.1: Windowing procedure. Finite-length windows $\mathbf{z}_k$ are extracted from the raw data stream $\mathbf{v}$ to obtain time-series.

TABLE 2.1: Summary table of the main adopted notation.

Symbol	Description
$t \in \mathbb{R}$	time
$\bar{t}$	time of interest
R	number of nodes in the power system cluster
$S \geqslant R$	number of data sources generating signals
$\bar{q} \geqslant R$	cardinality of raw signal
$\mathbf{v}(t) \in \mathbb{R}^{\bar{q}} \times \mathbb{R}$	raw signal
$k \in \mathbb{N}$	window index
$\tau \in [0, 1)$	window overlap parameter
n	cardinality of samples per window
$q \leqslant \bar{q}$	cardinality of pre-processed signal
$\mathbf{z}_k(t_{k,1}, \dots, t_{k,n}) \in \mathbb{R}^{n \times q}$	pre-processed signal
$p \leqslant n$	number of signal descriptors
$\mathbb{X} \subseteq \mathbb{R}^p$	domain of signal descriptors
$\mathbf{x} = x_1, \dots, x_p \in \mathbb{X}$	signal descriptors
N	number of observations available for training
$\{\mathbf{x}^1, \dots, \mathbf{x}^N\}$	set of input data for training
$\{y^1, \dots, y^N\}$	set of output labels for training
$D \in \mathbb{R}^{N \times (p+1)}$	design matrix for training
$\tilde{M}$	number of observations of reduced dataset (e.g. dictionary learning)
M	number of classes
$\mathbb{Y}$	class dictionary
$y \in \mathbb{Y}$	class label
$f(\bullet) : \mathbb{X} \to \mathbb{Y}$	classifier / association rule

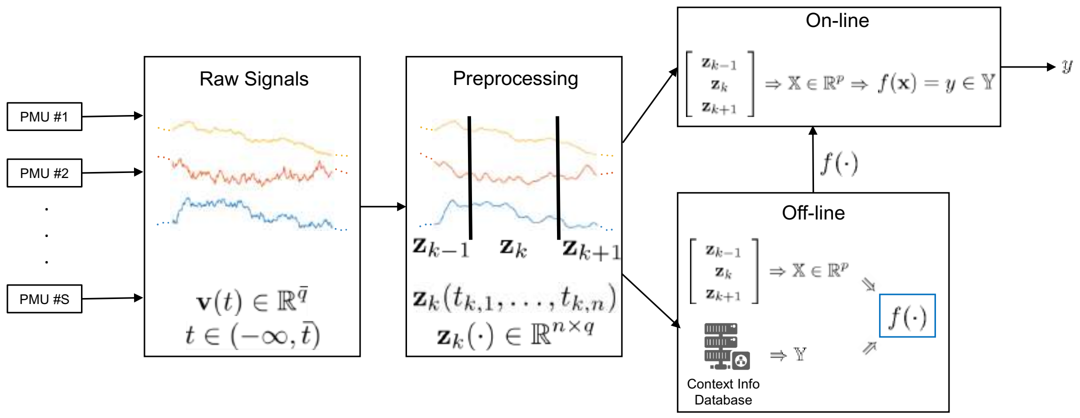

PMU #1
PMU #2
PMU #S
Raw Signals
$\mathbf{v}(t) \in \mathbb{R}^{\bar{q}}$
$t \in (-\infty, \bar{t})$
Preprocessing
$\mathbf{z}_{k-1}$ $\mathbf{z}_k$ $\mathbf{z}_{k+1}$
$\mathbf{z}_k(t_{k,1}, \ldots, t_{k,n})$
$\mathbf{z}_k(\cdot) \in \mathbb{R}^{n \times q}$
On-line
$\Rightarrow \mathbb{X} \in \mathbb{R}^p \Rightarrow f(\mathbf{x}) = y \in \mathbb{Y}$
y
$f(\cdot)$
Off-line
$\Rightarrow \mathbb{X} \in \mathbb{R}^p$
$\Rightarrow \mathbb{Y}$
Context Info Database

FIGURE 2.2: Data flow. Scheme of the data flow from sensors to classification.

2.1 The classification problem

The research on classification of time series has been of certain interest for some decades and in various fields, from speech recognition (Rabiner and Juang, 1993) to financial analysis (Tsay, 2005), from manufacturing (Susto and Beghi, 2016) to action recognition (Morris et al., 2014), and it is even more of key importance in this era of big-data and pervasive information flow. Specifically, two cornerstone issues need to be addressed:

- how do we compare different time-series? In particular, how do we compare time-series with different lengths?
- how can we recognize that different time-series are *realizations* of a common (unknown) process which represents a certain class?

The last question is particularly relevant in advanced monitoring applications: if a database of known failures is available, detection of current failures could be performed and exploited in predictive maintenance (Susto et al., 2013) / fault detection (FD) and FDI solutions . Some works formalize FD and FDI problems as semi-supervised ones, where particular classifiers (like One-Class-SVM) are built on a single group of data: such data are usually associated with normality conditions (Beghi et al., 2014); the goal of this classifiers is to create a solution that defines a 'normality space': when a new observation is available, it will be classified as anomaly if it lies outside the boundaries of the normality space. Such problem formulation can also be tackled with some of the methodologies presented in this chapter.

2.1.1 Classification methods taxonomy

For the sake of clarity, we provide a brief introduction to the different methodologies treated in this work that can be employed to solve the classification task with time-series data. Time-series classification techniques can be essentially divided into two main branches:

- **Feature-based (FB)**. FB methods perform a *feature extraction* procedure before the classification phase. Generally, from the original signal $\mathbf{v}(t)$ a moving window k of fixed length n is considered to obtain a time-series $\mathbf{z}_k$ and a set $\mathbf{x}$ of p *features* is calculated over it: to give some examples, commonly chosen features are mean, variance, maximum, minimum, entropy, all related to the time-series extracted from the signal.

  The idea underlying these methods is to capture signal statistics that identify a certain class of signals. In theory, if a process is weakly stationary then a second-order statistic is sufficient to characterize that signal; however, signals obtained from real-world scenarios are not stationary due to several nuisance factors and many more features may be necessary to summarize the informative content.

  In this respect, some observations are in order: unfortunately, non-automatic feature extraction procedures may be a time-consuming step that requires the involvement of process experts to avoid loss of information; moreover, extracted features designed to capture certain behaviors of the system, may not be informative under unseen conditions leading to poor monitoring performances. Finally, the tuning of n is far to be trivial for optimal results: normally, it is estimated through a cross-validation procedure (Friedman, Hastie, and Tibshirani, 2001). When dealing with the learning phase in FB methods, the learning rule is based on the definition of a dataset of N observations and of a *design matrix* as

$$D = \begin{bmatrix} \mathbf{x}^{(1)} & y^{(1)} \\ \mathbf{x}^{(2)} & y^{(2)} \\ \vdots & \vdots \\ \mathbf{x}^{(N)} & y^{(N)} \end{bmatrix} \in \mathbb{R}^{N \times (p+1)}. \tag{2.1}$$

- **Distance-based (DB)**. DB methods avoid the feature extraction phase in favor of the definition of suitable *distances*, among which the most common is Dynamic Time Warping (DTW) (Müller, 2007). Then, the classification phase is carried out through metric classifiers: one simple and popular choice and, surprisingly, one of the most effective is 1-Nearest Neighbor classifier (1-NN) (Friedman, Hastie, and Tibshirani, 2001).

  This strategy is motivated by the fact that the feature extraction phase could be time consuming and may cause significant loss of information about the original signal (Schirru et al., 2012). Conversely, though, due to nuisance factors, the DB direct comparison of time-series (e.g. by exploiting the Euclidean distance) may lead to ill-posed problems and unsatisfactory performances, thus calling for a careful selection of the distance metrics that trades off between complexity (of the measure) and accuracy (in the classification).

This main categorization is also summarized in Fig. 2.3.

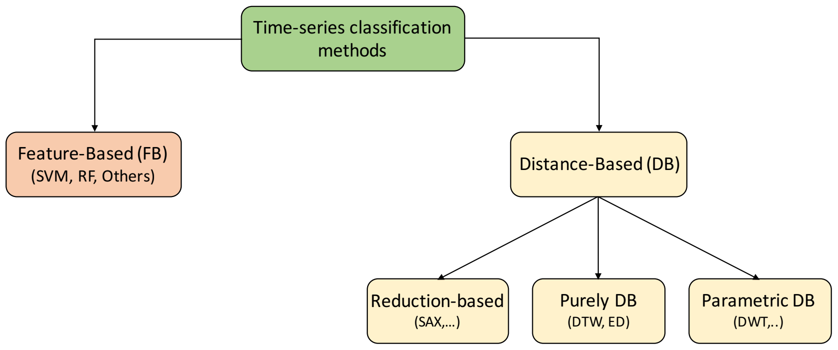

Time-series classification methods
Feature-Based (FB) (SVM, RF, Others)
Distance-Based (DB)
Reduction-based (SAX,...)
Purely DB (DTW, ED)
Parametric DB (DWT,..)

FIGURE 2.3: Classification taxonomy. Time-series classification methodologies tree highlighting the two families of feature-based and distance-based.

2.2 Data sources

For the sake of clarity, and to avoid confusion due to different notations used in related literature, we distinguish two data reduction techniques in which we are interested in: *dimensionality reduction* and *numerosity reduction*. In this Section we will present the most important time-series representations for these two types of reduction; for a more exhaustive review, we refer the interested reader to (Van Der Maaten, Postma, and Herik, 2009; Fu, 2011).

2.2.1 Dimensionality reduction

We refer to dimensionality reduction (DR) when dealing with FB techniques. More in detail, let us consider a design matrix $D \in R^{N \times p}$, with a high-dimensional feature space, e.g. when p is very large, (generally $p > 1000$); DR aims at finding a subset of informative features (*feature selection*) or, more generally, informative lower-dimensional structures through linear (e.g. principal/independent component analysis) and nonlinear (e.g. manifold learning) data transformation approaches.

In this context, although the space complexity can be overwhelming, the main issue is the renowned *curse of dimensionality* (Friedman, Hastie, and Tibshirani, 2001). It manifest itself in various ways, which all causes high variance and high bias of classifiers resulting in poor classification performance. In fact, when p is high, all feasible training samples sparsely populate the feature space and the concept of locality vanish. This problem, can be easily seen with the k-NN classifier and considering features uniformly distributed in a p-dimensional unit hypercube: when $p = 1000$, in order to capture for example 10% of data to evaluate local average, it is necessary to consider the 99.7% of the range of each feature.

DR techniques review

As already stated above, the informative content of a signal is often embedded in a low-dimensional space that can be isolated through DR techniques. Among the most popular methodologies, we mention here Generalized Discriminant Analysis (Scholkopft and Mullert, 1999), Independent Component Analysis (ICA) (Hyvärinen, Karhunen, and Oja, 2004; Subasi and Gursoy, 2010), Kernel PCA (Schölkopf, Smola, and Müller, 1997; Mika et al., 1998), Linear Discriminant Analysis (LDA) (McLachlan, 2004), Multi-Dimensional Scaling (MDS) (Kruskal and Wish, 1978) and Principal Component Analysis (PCA) (Hotelling, 1933; Jolliffe, 2002). In the realm of non-linear approaches, manifold learning, whose objective is to learn the hidden manifold described by the data (Tenenbaum, De Silva, and Langford, 2000; Roweis and Saul, 2000; Cayton, 2005; Narayanan and Mitter, 2010), has been gaining lot of attention in the past recent years.

A simpler, but often equally efficient, DR approach is to remove redundant (correlated) features, selecting a subset of relevant features, instead of finding the underlying low-rank structure of the data at hand. Well-known feature selection techniques are backward feature elimination, forward feature construction, minimum-Redundancy-Maximum-Relevance (mRMR), just to provide some examples. Interestingly, Random Forests (RFs), beside being amongst the most effective classifiers, are also powerful instruments for feature selection (Biau, 2012).

For an exhaustive description, we refer the interested readers to (Tang, Alelyani, and Liu, 2014).

2.2.2 Numerosity reduction

We refer as numerosity reduction (NR) when aiming at reducing data volume by choosing alternative, smaller forms of data representation of the signals at hand. It differs from DR in the sense that it aims at finding a different representation of time-series and/or reducing the number N of training examples needed for classification without reducing accuracy. For example, consider a collection of N input univariate time-series $\{\mathbf{x} \in \mathbb{R}^n\}_{i=1}^N$. Data reduction techniques aim at reducing n and/or N.

NR techniques review

In this framework, to reduce both n and N, parametric and non-parametric approaches may be employed for NR. Parametric approaches model the time series using a parametric model such as Discrete Wavelet Transform (DWT), Discrete Fourier Transform (DFT) and Log-linear models to cite the most common examples. Then the complexity space reduce from $O(n)$ to $O(p)$ where $\mathbf{x} \in \mathbb{R}^p$ is p-dimensional vector of parameters of the model. An example of non-parametric approaches is histogram or, simply, sampling.

Another family of NR approaches is *symbolic representation*, for which Symbolic Aggregate Approximation (SAX) (Lin et al., 2003a; Lin et al., 2007) is the most know technique. SAX technique mainly consists on 3 phases:

- signals standardization in order to obtain a zero mean and unit variance signal;
- Piecewise Aggregate Approximation (PAA) (Keogh et al., 2001), described in the following;
- symbolic mapping through discretization on amplitude domain.

After normalization, in the PAA phase, a signal $\mathbf{z} = z_1, \dots, z_n$ let $\bar{\mathbf{z}} = \bar{z}_1, \dots, \bar{z}_p$ of length s is discretized on time in p frames in order to obtain a vector $\bar{\mathbf{z}} = \bar{z}_1, \dots, \bar{z}_p \in \mathbb{R}^p$. Formally, the resulting i-th element $\bar{z}_i$ is defined by the mean of i-th interval:

$$\bar{z}_i = \frac{p}{s} \sum_{j=\frac{s}{p}(i-1)+1}^{\frac{s}{p}i} z_j \tag{2.2}$$

Then, the SAX representation procedure (i.e. the discretization on amplitude domain) can be summarized as follows. Let a_i denote the i-th element of the alphabet $\mathcal{A}$, with $|\mathcal{A}| = \alpha$. The mapping from the PAA approximation to the correspondent word $\mathbf{x} = x_1, \dots, x_p$ of length p is obtained as follow:

$$x_i = a_j \qquad \text{iif} \qquad \beta_{j-1} \leqslant \bar{z}_i < \beta_j, \tag{2.3}$$

where $\{\beta_j\}_{j=1}^{\alpha-1}$ are break-points tuned to have symbols with equiprobable occurrence. One of the advantages of introducing the *SAX representation*, is that a new distance measure - which is a lower bound of euclidean distance - can be immediately defined. Let $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ be two time-series of same length n and $\mathbf{x}^{(\mathbf{1})} = x_1^{(1)}, \dots, x_p^{(1)}$ and $\mathbf{x}^{(\mathbf{2})} = x_1^{(2)}, \dots, x_p^{(2)}$ be their *SAX symbolic representation*; the SAX distance is defined as:

$$D_{SAX}(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = \sqrt{\frac{n}{p} \sum_{i=1}^{p} dist\left(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}\right)^2}. \tag{2.4}$$

Another popular symbolic approach is the Symbolic Fourier Approximation (SFA) (Schäfer and Högqvist, 2012). The SFA accepts the same parameters p and α as SAX. In this case, p represents the number of Fourier coefficients (real and imaginary) used. Naturally, each sliding window is normalized to have a standard deviation of 1 to obtain amplitude invariance, before applying SFA. Provided the parameters, the SFA symbolization is carried out in two main steps:

1. pre-processing phase called Multiple Coefficient Binning (MCB) discretization,
2. SFA transformation.

In the phase 1) the p coefficients (real and imaginary) c_i are extracted for all the training time-series and histogram is built for each c_i, where each bin corresponds. Then each histogram is used to infer the breakpoints β_{ij}, $i = 1, \ldots, wp$, $j = 1, \ldots, \alpha + 1$ in order to make symbols equiprobable. In the phase 2) each coefficient is extracted and it mapped to a symbol according to breakpoints found in the MCB phase. Then the string representing the time-series is formed by the sequence of coefficients. Thus, to find a second-order resolution (2 coefficients) there are 2 real plus 2 imaginary coefficients resulting in a word of length 4. SFA presents some important difference from SAX: First of all, the time complexity (to transform a single time-series $\mathbf{z}$) is $O(n \log n)$ while SAX time complexity is $O(n)$. However, provided the same word length, SFA best represents the raw signal as it does not apply any piece-wise discretization, but expresses a linear combination of continuous Fourier functions through the learned coefficients. Moreover, in opposite to SAX, if we wanted a finer resolution increasing p, it would not be necessary to recalculate all DFT coefficients as the symbols of a smaller word length are always a prefix of the larger word lengths.

Regarding the reduction of N, the most popular approaches are clustering and dictionary learning. Time-series clustering aims at finding groups (clusters) in which data can be divided. A way to speed up clustering approaches (that are generally in the realm of *lazy learning* approaches (Friedman, Hastie, and Tibshirani, 2001)), "mean" (or centroid) time-series for each cluster are usually taken as representative of each group: this generally require $\tilde{M}$ comparisons instead of N, and, since $N \gg \tilde{M}$, this decreases considerably the time to perform the clustering. Similarly, dictionary learning techniques find a sort of base of $\tilde{M}$ signals, from which a given signal can be represented as a linear combination. Indeed, in the classification context, the *supervised* dictionary learning aims at learning a dictionary containing the elements which best represent the classes and thus they find a *discriminative* representation.

2.3 Classification methods

2.3.1 Feature-based methods

Assuming a dataset of N time-series $\mathbf{z}_k, k = 1, \dots, N$, and M possible target classes $\mathbb{Y} = \{y_1, \dots, y_M\}$ that jointly describe the classification problem of interest, FB methods focus on finding a compact description $\mathbf{x} = [x_1, \dots, x_p] \in \mathbb{X}$ of the time-series $\mathbf{z}_k$ such that $p \leqslant n$ (and typically $p \ll n$); all these N observations are collected into a matrix $D \in \mathbb{R}^{N \times (p+1)}$, called *design matrix*, as defined in (2.1) in Sec. 2.1.1. In practice, D represents the supervised learning phase of the procedure and it is exploited to define the rule $f(\bullet) : \mathbb{X} \rightarrow \mathbb{Y}$ according to a chosen classification method (Friedman, Hastie, and Tibshirani, 2001) as better detailed in the following.

In this category, the most employed classification algorithms are k-Nearest Neighbor (k-NN) (Friedman, Hastie, and Tibshirani, 2001), Support Vector Machines (SVM) (Boser, Guyon, and Vapnik, 1992; Cortes and Vapnik, 1995), Relevance Vector Machine (RVM) (Tipping, 2001), Decision Trees (DT) (Breiman et al., 1984), Random Forest (RF) (Breiman, 2001), Logistic Regression (LR) (Cox, 1958), Gaussian Processes (GP) (Rasmussen, 2006), and Deep Neural Networks (DNN) (Goodfellow, Bengio, and Courville, 2016a). We refer the interested reader to the literature references for further details on the specific methods and the general textbooks (Friedman, Hastie, and Tibshirani, 2001; Bishop et al., 2006).

Metrics based approaches

The techniques above are not straightforwardly applicable to continuous time-series since they need input vectors of fixed length, and this justifies their inclusion in the FB methods[1]. As a matter of fact, even if the time-series in input to the classifier were discrete-time and of fixed length, the accuracy performance would be poor due to two main reasons: firstly, it is common to consider long sequences of samples of $n > 100$ or even $n > 1000$; in these cases the space spanned by the time-series is too large and sparse incurring in the aforementioned "curse of dimensionality" problem. Secondly, considering the discrete values as independent features would be not reasonable, since they do not provide any information *per se* about the characteristics of the signal, since time-series values are strongly highly correlated on time and the feature extraction phase is exactly designed so as to highlight this correlation.

The flowchart of the FB classification procedure is given in Fig. 2.4: after a pre-processing phase on the raw signal characterized by the presence of a low-pass filtering operation to reduce measurement noise and the windowing procedure, the proper feature extraction task is performed on the time series, resulting in a compact set of signal descriptors, which undergo the classification phase.

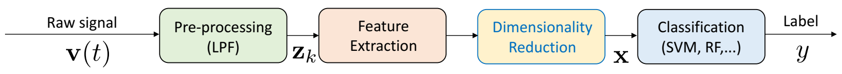


FIGURE 2.4: FB methods. Operation flow of FB classification procedures.

The main advantages of the FB methods clearly reside in the the compactness in the representation able to characterize the signal. Typical examples of features are:

[1]This is partly true for Deep Learning approaches: recent Deep Neural Networks schemes can avoid the feature extraction phase.

- sample features: sample variance/mean/RMS value of the time-series;
- energy/power features: energy value from the Discrete Fourier Transform (DFT) coefficients, power spectrum bands;
- correlation features: number/location/width of (prominent) peaks in the autocorrelation function (repetitive and periodic signals present a peak in the autocorrelation function); correlation parameters among different signal dimensions.

Unfortunately, though, the FB approach presents also several drawbacks:

- features must be defined ad-hoc depending to the task[2];
- high dimensionality;
- non-stationarity of time-series;
- time structures are not considered.

It clearly appears from these lists that most of feature-based ML techniques are not adapted to exploit time structures, i.e. patterns, which are an *intrinsic* and *distinctive* characteristic of each time-series. In this regard, the first attempts to exploit patterns can be found in Manganaris (1997), Kadous (1999), and Kudo, Toyama, and Shimbo (1999). In particular, in (Kadous, 1999) parametrized events are extracted from multivariate time-series: these events are clustered in the parameters space and the resulting prototypes are used as basis to build classifiers. Instead, Kudo, Toyama, and Shimbo, 1999 maps multivariate time-series into binary vectors: the space value-time is represented by a grid and each element (of the vector) is associated to one cell of the grid count: if the signal passes through the corresponding cell, the element is `true (1)` otherwise it is `false (0)`. Then, these converted binary vectors are used as the basis for the classification.

More in general, a limitation to these methods stems from the fact that classification rules are extracted taking into account absolute time values, leading to the inability of handling situations where particular behaviors happen at different time values: Geurts, 2001 argues that many time-series classification problems can be solved by detecting and combining local properties (patterns) on time-series, and proposes a procedure that captures the information of shift-invariant patterns using DTs over piece-wise constant time-series.

More recently, *Interval Features* (IFs) have been introduced in order to capture temporal information (Rodríguez, Alonso, and Boström, 2001; Rodríguez and Alonso, 2004; Rodríguez, Alonso, and Maestro, 2005; Deng et al., 2013). These features are common statistics such as mean, variance, slope but they are calculated over random *intervals* exploiting a boosting procedure. The first idea on IFs was presented in Rodríguez, Alonso, and Boström, 2001, later expanded by Rodríguez and Alonso, 2004; Rodríguez, Alonso, and Maestro, 2005 using classifiers such as DTs and SVMs applied on the features extracted from binary ensembles. However, as discussed in Deng et al., 2013, the number of candidate splits is generally large and thus there can be multiple splits having the same ability of separating the classes. To cope with these issues, Deng et al., 2013 introduces an additional measure able to better distinguish among IFs. Another problem in boosting IFs is the size of the relative

[2]This is one of the main reasons that favors the usage of deep learning in complex problem such as natural language processing and computer vision: in these fields, the definition of informative features has required at least 20 years of research.

space that is $O(n^2)$, where n is the length of a time-series. In Rodríguez, Alonso, and Boström, 2001 the feature space is reduced to $O(n \log n)$ using only intervals of length equal to powers of two. Deng et al., 2013 consider the same approach of random sampling strategy used on RF (Breiman, 2001) further reducing the feature space at each node to $O(n)$.

Occurrence counting approaches

Another type of approaches to classify time-series is the so called Bag-of-Words (BoW), also Bag-of-Features (BoF), nowadays used in image classification and document classification and classically developed in the context of natural language processing. BoW consists in representing data (images in computer vision or documents in natural language processing) using a histogram of word occurrences, where a *word* is a task-dependent element (Lowe, 1999), namely a proper textual word in language processing or the image description through intensity local gradients in computer vision. After this encoding, the classification task is reduced to computing a histogram-based similarity (typically using euclidean distance).

With respect to the two steps of (i) conversion of the time-series into a BoW representation (that is, an histogram of the word occurrences) and (ii) training of a classifier (such as SVM, RF, kNN) upon BoW features, we particularly focus on step (i) in the following, since step (ii) is similarly performed by all methods using RF, SVM or some other common classifier over the word histogram. Indeed, several BoW-inspired techniques have been recently investigated in order to extract local and global features (Lin, Khade, and Li, 2012; Wang et al., 2013; Baydogan, Runger, and Tuv, 2013; Senin and Malinchik, 2013; Baydogan and Runger, 2015; Bailly et al., 2015; Schäfer, 2015b; Schäfer, 2015a; Bailly et al., 2016). Hereafter, a brief overview of the main contributions in given.

In the computer vision field, the BoW technique is used for image classification (Csurka et al., 2004) often in combination with the Scale Invariant Feature Transform (SIFT) technique. SIFT is a *co-variant* detector, which extracts local features (keypoints) that are robust to noise (e.g. changes in illumination) and invariant to affine transformations and scale. As a general note, BoW methods ignore temporal ordering, which may cause that patterns in observed time-series or images are not identified. Nonetheless, some BoW-based works try to indirectly remove this lack, although to a limited extent. For example, in Bailly et al., 2015; Bailly et al., 2016 a variant of SIFT for time-series is applied and local features (key-points descriptors) are extracted with a procedure very similar to SIFT and BoW approach is used over the SIFT descriptors. This choice is motivated by the fact that SIFT captures local structures while BoW allows to describe the global behavior of the time-series. Furthermore, in Bailly et al., 2016, the same authors adopt dense-SIFT-like descriptor: the main difference with the previous work is that keypoints no longer correspond to extrema but are rather extracted at all scales every time step on Gaussian-filtered time series. This approach in general leads to more robust global descriptors, especially when local extrema can be found (when signal are very smooth).

In Wang et al., 2013, Discrete Wavelet Transform (DWT) is applied on sliding windows of the time-series and the resulting DWT coefficients form a word for each window (segment). In the training phase all the DWT segments are clustered through k-means in order to obtain a word dictionary $\mathbf{D}$. In the classification phase each DWT window is assigned to the nearest word in $\mathbf{D}$, to build a histogram that is used to computed the similarity; the classification is finally carried out using 1-NN.

A bag-of-features framework is proposed also in Baydogan, Runger, and Tuv, 2013, which combines interval features and BoW. Here, there are extracted interval features and start/end time points over random subsequences of random length, and a supervised codebook of class probability estimate (CPE) histogram is built in the training phase: for each sequence (of the time-series) a CPE is found using a RF classifier. Then, all the CPEs are quantized in order to form a different histogram for each class, which are concatenated into a single histogram of each time-series and are used as features in combination with other global features. Finally, the employed classifier is RF.

A symbolic multivariate time series (SMTS) method is discussed in Baydogan and Runger, 2015: each time-series is represented by a feature space which contains the time instants, the time-series values and the first difference values, all collected in a design matrix D. Then D is input to a symbolic discretization which is obtained using tree-based classifiers (supervised discretization). Then the classification is performed using a common BoW approach based on histograms of symbols. Its total computational complexity is due to the number of trees, the number of training instances and the number of time-series subsequences extracted. Then, multivariate time-series are mapped into a feature matrix, where features are vectors containing a time index $\bar{t}$, the values and the gradient of time series at t for all dimensions. The feature space is finally partitioned into regions (that is, symbols) by a RF classifier. An appreciable properties of SMTS is that it does not requires tuning parameters, while one main drawback of this representation is the possibly high dimensionality, which limits its application for large datasets.

In Schäfer, 2015b the Bag-of-SFA-Symbols (BoSS) is introduced. An univariate time-series $\mathbf{z}$ is represented by SFA words and then an histogram is built. However, since this approach is $O(N^2n^2)$ for training, $O(Nn)$ for classification, in Schäfer, 2015a it is presented a "scalable version" named BoSS-VS that uses vector space models. In this case, once the BoSS histogram is obtained, for each SFA word w, the word (called *term*) frequency tf of w in a certain class c_i is computed, together with the ratio idf given by the total number of classes divided by the number of classes in which w appears. Then, the $tf - idf$ measure is obtained by the product $tf \cdot idf$: this measure is used to weigh the word frequencies in the vector to give a higher weight to representative words of a class. The motivation behind this choice is that an high $tf - idf$ for a word w means that w appears with an high-frequency in a specific class c_i, while low $tf - idf$ values means that w in common in all classes. When a new observation $\mathbf{z}_{new}$ arrives, the BoSS histogram and the relative tf vector are computed. Then the *cosine similarity*[3] is obtained in order to predict the nearest class. Using this model (named "term frequency inverse document frequency", $tf - idf$, model), the complexities of training and classification reduce to $O(Nn^{3/2})$ and $O(n)$, respectively.

In Lin, Khade, and Li, 2012; Senin and Malinchik, 2013 time-series are mapped into SAX words through a sliding windows partitioning, which are used to build histograms of n-grams: for each time-series an histogram counts the frequency of occurrences of each SAX word and thus each time-series is represented by its histogram. In particular, in Senin and Malinchik, 2013, the authors combine SAX and Vector Space Model (VSM) (Salton, Wong, and Yang, 1975) exploiting the $tf - idf$ model (Luhn, 1957; Sparck Jones, 1972), weighing bags and cosine similarities as metrics. This technique has a parameter space of dimension $O(n^2)$ and needs to recompute all SAX coefficients for each new choice of parameters p (number of

[3]Given two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, both in $\mathbb{R}^p$, the cosine similarity is defined as $\cos\beta = \dfrac{\langle \mathbf{x}_1, \mathbf{x}_2 \rangle}{\|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|}$, where $\langle \bullet, \bullet \rangle$ is the inner product.

frames in the PAA representation) and α (cardinality of the alphabet); moreover, the training time is $O(Nn^3)$ where N is the number of training instances.

Dynamics based approaches

The last set of FB techniques we present in this review explicitly takes into account the *dynamics* of the signals and comes from *dynamical systems* and *signal processing* theory. In the context of dynamical systems and identification theory, in the past decades much attention has been conveyed on modeling stochastic processes (whose realizations are time-series) in order to *predict* their future trends and values. The most common models are Auto-Regressive (ARV), Auto-Regressive-Moving-Average (ARMA), Auto-Regressive-Integrated-Moving-Average (ARIMA) models (Ljung, 1998), just to give some examples. With such approaches, the coefficients themselves of the fitted model are used as features for a suitable classifier (Garrett et al., 2003; He and Jin, 2008) or are used to build a more complex generative model (Roberts, 2002).

In more detail, Roberts, 2002 takes a Bayesian point of view and proposes a hierarchical model that consists of a feature extraction stage and a generative classifier, probabilistically linked by a latent feature space. The classifier is implemented as an Hidden Markov Model (HMM) with Gaussian and multinomial observation distributions defined on a representation of ARV models. The HMM is used to model the correlation between adjacent windows (subsequences), that is, this model assumes that time-series are consecutively extracted from an unique flow.

In a similar way, signal processing transformations such as DFT or DWT are applied to the raw signals to obtain coefficients that can be exploited in training suitable classifiers (Jahankhani, Kodogiannis, and Revett, 2006; Subasi, 2007; Übeyli, 2009). Interestingly, DWT results to be more suitable for non-stationary time-series and, conversely w.r.t. DFT, is ideal for identifying highly discriminant local time and scale features (Huang et al., 1998a). We note that DWT and DFT and dynamical models can be seen as dimensionality reduction procedures: in the next section, we will revise these concepts from the point of view of distance-based methods.

Eruhimov, Martyanov, and Tuv, 2007 gathered the most known features deriving from the presented methods such as statistical moments, wavelets coefficients, PCA coefficients, Chebyshev coefficients, and the original values of time-series and built a classifier from them. However this method can be accurate at the cost of complexity and a feature selection procedure is needed to reduce the dimensionality.

Although time-pattern (dynamic) information has been considered in literature, most of feature-based models present common limitations due to the nature of time-series. In fact, the presence of variability in the time-series causes these methods to be ineffective to cope with common issues. The variability arises because of the stochasticity of the process generating the time series, non-stationarity of time series and nuisance factors. To give a practical example, the most effective FB methods that exploit time-patterns presented in this chapter are not able to deal with variable time-series lengths and the other methods which can handle this issue exploits only global statistics making them ineffective with non-stationarity.

Finally, perhaps the most limiting issue of feature-based methods is that the feature extraction phase can be demanding both in terms of memory and computational burden. These factors could make feature-based methods not suitable for resource-constrained devices. In Table 2.2 the main peculiarities of each FB methods are summarized.

TABLE 2.2: Main characteristics of feature-based methods which have been reviewed in this work.

Methods	Characteristics
Time-pattern features	First attempt to capture local structures
Interval Features	Capture temporal information
	Feature space is big ($O(n)$)
	Feature Extraction can be onerous
Bag-of-Features	Local and global structures are captured
	Histogram extraction is onerous
	Time-patterns not considered
Dynamic Features	Encode information about dynamic
Frequency Domain Features	Capture behavior in the frequency domain
	FFT/DWT are efficiently implemented
Ensemble of features	Good accuracy
	Feature extraction is very onerous

2.3.2 Distance-based methods

DB methods can be clustered into three groups:

- *purely distance-based* - These methods are based on the direct computation of ad-hoc defined distances over raw time-series.
- *reduction distance-based* - Such methods are based on the computation of opportunely defined distances over a reduced representation of raw time-series.
- *parametric distance-based* - With this type of DB approaches, raw signals are represented with a combination (generally linear) of basis signals (e.g. sine functions in the Fourier Series representation). The coefficients of different representations (parameters) are used for the computation of ad-hoc defined distances.

A general picture of the dataflow for DB methods is given in Fig. 2.5 and the three groups will be discussed in detail in the following of this section.

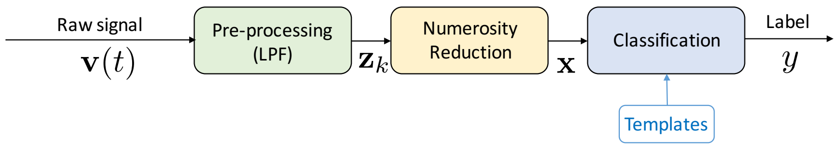


FIGURE 2.5: DB methods. Operation flow of DB classification procedures.

Purely distance-based methods

Purely DB methods perform the classification task by adopting a classifier that exploits an opportunely defined distance applied to the raw time-series $\mathbf{z}$. Thus, in this case, we have $p \equiv n$ and we refer to $\mathbf{x}$ as the time-series $\mathbf{z}$ (the map $\mathbf{z} \rightarrow \mathbf{x}$ can be seen as the identity map). Here, we consider a set of variable-length training time-series and the corresponding label $D = \{(\mathbf{x}_i, y_i), i = 1, \ldots, N\}$. As mentioned earlier, purely DB methods are based on the computation of a distance over the raw time-series. In choosing a distance, the most straightforward approach is to adopt an

Euclidean distance, however, this choice has many drawbacks: Euclidean distance to be computed requires time series of equal lengths; moreover, even when comparing two series of equal length, Euclidean distance can be an unfortunate choice since it does not consider common nuisance factors such as warping (Batista, Wang, and Keogh, 2011).

For the previous reasons, a more popular approach for distances is Dynamic Time Warping (DTW) (Sakoe and Chiba, 1978). DTW measures the similarity between two time-series with, possibly, different lengths by warping the time axis of one (or both) sequences to achieve alignment between the two. DTW provides a *similarity score*, an index on how similar two time series are: in order to define the similarity score, let's consider two time series $\mathbf{x}^{(1)} = \{x_1^{(1)}, \dots, x_n^{(1)}\}$ and $\mathbf{x}^{(2)} = \{x_1^{(2)}, \dots, x_m^{(2)}\}$ and let us define a grid $\mathcal{G} = [n] \times [m]$. A *warping path* wp in $\mathcal{G}$ is a sequence $wp = (\mathbf{p}_1, \dots, \mathbf{p}_l)$ with points $\mathbf{p}_k = (i_k, j_k) \in \mathcal{G}$ s.t.:

$$\mathbf{p}_1 = (1,1) \quad \text{and} \quad \mathbf{p}_l = (n,m) \qquad \text{(boundary conditions)}$$
$$\mathbf{p}_{k+1} - \mathbf{p}_k \in \{(1,0),(0,1),(1,1)\} \qquad \text{(warping conditions)}$$

$\forall k | 1 \leqslant k < l$. The cost of 'warping' $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ along the warping path s is given by

$$d_s(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \sum_{(i,j)\in s} (x_i^{(1)} - x_j^{(2)})^2, \tag{2.5}$$

where $(x_i^{(1)} - x_j^{(2)})^2$ is called *local transformation cost*. Then, the DTW similarity score is defined by

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \min_s d_s(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) \tag{2.6}$$

Regarding the choice of the classifier, the most common choice is 1-Nearest Neighbour (NN) combined with DTW (NN-DTW). Notably, even if NN is generally considered as one of the simplest approaches to classification, in many papers NN-DTW outperforms more sophisticated approaches when dealing with time-series classification (Xi et al., 2006; Batista, Wang, and Keogh, 2011; Lines and Bagnall, 2015). One of the issues of NN-DTW is its computational cost: the DTW is $O(n^2)$ and it has to be evaluated for each training example in order find the nearest neighbor. In order to alleviate the aforementioned issue, various approximations of DTW (Keogh and Ratanamahatana, 2005; Salvador and Chan, 2007; Al-Naymat, Chawla, and Taheri, 2009; Candan et al., 2012; Silva and Batista, 2016) have been introduced: the most promising are FastDTW (Salvador and Chan, 2007) and SDTW (Candan et al., 2012). FastDTW adopts a multi-scale approach that recursively projects a solution from a coarse resolution and refines the projected solution. FastDTW time and space complexity are $O(n)$; SDTW, instead, extracts keypoint descriptors (similarly to SIFT (Lowe, 1999), a popular approach in computer vision) and uses them to reduce complexity.

Beside approximations, several other extensions and modifications of DTW have been proposed: Keogh and Pazzani, 2001 proposed the Derivative Dynamic Time Warping (DDTW) which transforms the original time-series into a first order differences time-series, in order is to avoid ill-conditioning; ill-conditioning is a common issue in DTW when dealing with noisy and long signals due to the fact that single points of one of the compared time-series could be mapped onto a large subset of the other time-series leading to poor alignments. Jeong, Jeong, and Omitaomu, 2011 proposed a penalty-based DTW (WDTW), which adds a multiplicative weight penalty in order to penalize points with higher phase difference between a reference point and a testing point. This, has the aim to prevent minimum distance distortion caused by outliers. Other used similarity measures are Edit Distance with Real Penalty (ERP), proposed by Chen and Ng, 2004; Chen, Özsu, and Oria, 2005 and Time Warp Edit Distance (TWED) proposed by Marteau, 2009. ERP is a variant of L1-norm, which can support local time shifting. It can also be viewed as a variant of EDR Chen and Ng, 2004 and DTW, but it is a metric distance function. TWE distance is an elastic distance measure (efficiently implemented using dynamic programming) which, unlike DTW, is also a distance. It allows warping in the time axis and combines the edit distance (defined for time-series) with L_p-norms. Marteau, 2009, also provides a lower bound for the TWED measure which allows to operate into down-sampled representation spaces in order to fasten the algorithm.

As we already states, DTW is not a distance measure and this implies that it cannot be employed with kernel methods (Scholkopf and Smola, 2001), where kernel must be positive definite. Moreover, time-series of different length cannot be compared. In this context, Cuturi et al., 2007 proposes a new family of kernels between variable-length time series, called Alignment Kernels, which consider the soft-max of the score of all possible DTW-based alignments to consider the 3 of the scores of all possible alignments. However, the computation of such kernels can be performed in quadratically, and motivated by this limitation, Cuturi, 2011 provides an efficient version of it.

Although the usage of kernel methods combined with global alignment, allows to consider variable-length time-series and disturbances which cause time warping, these do not consider the *dynamics* or patterns. Indeed, albeit the term "dynamic" (deriving form dynamic programming), DTW has nothing which considers the *dynamics* of time-series we want to classify. Soatto, 2007, among other contributes on defining distances for non-stationary time-series, also introduces the Dynamic Time Warping Under Dynamic Constraints (DTWUDC), which constrains the DTW to follow a dynamical system. More in detail, in Soatto, 2007 it is assumed that the data (of two time-series) are outputs of dynamical models driven by inputs that are warped versions of some common function. Thus, given two univariate time series $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ $(i = 1, 2)$, he assumes that there exists the dynamical model (linear in the parameters)

$$\begin{cases} \dot{h}_i(t) & = Ah_i(t) + Bu(w_i(t)) \\ \mathbf{x}^{(i)}(t) & = Ch_i(t) + n_i(t) \end{cases}$$

where A, B, C are suitable matrices, h_i are the state functions, $n_i(t)$ are noise processes, $w_i(t)$ are warping functions, u is a common input and $\mathbf{x}^{(i)}$ is the time-series i. Then, the distance can be evaluated in two stages to fit $u_i(t)$ and then $w_i(t)$. In the subsection dedicated to Parametric distance-based methods we will see other methods, mainly kernel methods, on dynamical systems which capture the dynamic essentials of the time-series.

Reduction distance-based methods

As we detailed above, although the 1-NN-DTW classifier is remarkably difficult to beat, it presents computational issues which prevent its usage in resource-constrained systems. Moreover, shape-based methods typically fail to provide satisfactory results for long time-series, as the weight of discriminative “local” structures decreases. In this context, albeit various approximations of DTW was presented, the 1-NN still remains a bottleneck, as it requires the comparison with all the training time-series, mostly when the length n and the number N of the time-series are large. Moreover, it also requires space to store the entire dataset, which is unfeasible for most resource-constrained devices.

As we have seen in Sec. 2.2, data reduction techniques aim at reducing n and/or N. The main idea of Reduction distance-based methods is to reduce the time-series in a parsimonious representation space and compute a suitable distance in this space. In this part, we will discuss distance-based methods which reduce n or N or both. Before reviewing these techniques, we notice that some of the techniques we have previously exposed, may also fall into this category. For example, this is the case of VS-BOSS, in which the time-series $\mathbf{x}$ is mapped into histograms and the histogram-similarity is computed in order to classify $\mathbf{x}$.

As we saw previously, symbolic representations such as SAX are very useful for numerosity reduction. In this context, the simplest classifier is the 1-NN-SAX classifier, that is the 1-NN on the space of symbolic representation endowed with the metric d_{SAX}. However, as the Euclidean distance, it is not robust to time distortions or more simply time shifts. Moreover, as we pointed out, when N is large, using SAX approximation may not be enough. Thus, it has been crucial to find some sparse representation of the space of training examples.

The rationale under the reduction of N is to find a subset of canonical $k \ll N$ examples (*templates*) which best describe training set without loss of information (in the sense that they are *sufficient*). Fundamentally there are two directions to find this templates, that is, unsupervised and supervised. In the unsupervised approach, *templates* are found trough clustering techniques regardless the task at hand. On the other hand, supervised approaches aim at finding also the most discriminative templates for the classification tasks, i.e. the templates which best represent each class. Naturally, supervised methods are most suited for the classification task. In literature, there are two (at least) different definitions of template, that is *shapelet* and *dictionary*. They rely on the same idea, but is tackled with different approaches.

The concept on *shapelet* has been developed in the recent literature (Ye and Keogh, 2009; Ye and Keogh, 2011; Mueen, Keogh, and Young, 2011; Lines et al., 2012; Rakthanmanon and Keogh, 2013; Hills et al., 2014; Grabocka et al., 2014). Shapelets are sub-sequences of time-series which are maximally (in some sense) representative of a certain class and thus are useful to classify unlabeled time series; shapelets are maximally representative in the sense of the *information gain* criteria also used to train decision trees and RFs (Breiman et al., 1984).

The main advantages of using shapelets, is that 1-NN with all the training instances is avoided in favor of the computation of the distance to the shapelets which represent each class and it is phase-invariant contrary to simpler techniques such as 1-NN with Euclidean Distance (or SAX distance). On real problems, the speed difference of classification can be greater than three orders of magnitude (Ye and Keogh, 2009). However, despite the fast classification, the training of the shapelets is onerous. In the first work where shapelets for classification was introduced, the worst-case scenario for the training time was $O(N^2n^3)$ where N is the number of time series in the dataset and n is the length of the longest time series in the dataset. In order to reduce the training complexity, various extensions have been proposed. Among all, in Rakthanmanon and Keogh, 2013, SAX is used to find sub-optimal shapelets reducing training complexity to $O(Nn^2)$: in this case, time-series are mapped to a low dimensional space of SAX words and shapelets are found directly on this space. Then the distance used for classification is the d_{MAX} defined above.

Dictionary learning

The other approach we find in the literature is called *dictionary learning*, whose aim is to learn a sparse representation of the time-series in terms of a basis of signals and express the input signals as a linear combination of basic elements belonging to a set called dictionary. Before continuing we notice that we refer to $\mathbf{x}$ as the time-series $\mathbf{z}$, that is $p = n$. Dictionary learning can be categorized in two approaches: unsupervised and supervised. In order to understand their difference, in the following we briefly present the main concepts of these frameworks which are detailed for example in Mairal et al., 2008; Mairal et al., 2009; Mairal, Bach, and Ponce, 2012. Suppose of having N univariate fixed-length training time-series $\{\mathbf{x}^{(i)} \in \mathbb{R}^n\}_{i=1}^N$ associated to binary labels $\{y_i \in -1, +1\}_{i=1}^N$. In order to find an optimal (in the sense of Mean Square Error) and sparse representation, through dictionary $\Delta = [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_{\tilde{M}}]$ of signal $\mathbf{x}$ we can solve the convex optimization problem

$$\min_{\boldsymbol{\alpha}, \Delta} \sum_{i=1}^{N} \|\mathbf{x}_i - \Delta \boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1, \qquad s.t. \quad \|\boldsymbol{d}_i\|_2 = 1, \tag{2.7}$$

where ℓ_1-norm is used for $\boldsymbol{\alpha}$ since encourages sparsity Friedman, Hastie, and Tibshirani, 2001 inducing non-informative $\boldsymbol{\alpha}_i$ to be zero. Once obtained optimal $\boldsymbol{\alpha}^*, \Delta^*$, we can solve the classification task solving

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha}^*, \Delta^*, \boldsymbol{\theta})) + \lambda_2 \|\boldsymbol{\theta}\|_2^2 \tag{2.8}$$

where $L(\cdot, \cdot)$ and f are an opportune loss function and the predicting function which together define a classifier and $\boldsymbol{\theta}$ parametrizes the model f. Common choices of f are:

1. linear models in $\boldsymbol{\alpha}$: $f(\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\theta}) = \boldsymbol{w}^T \boldsymbol{\alpha} + b$ with $\boldsymbol{\theta} = \boldsymbol{w} \in \mathbb{R}^k, b \in \mathbb{R}$
2. bilinear models in $\mathbf{x}$ and $\boldsymbol{\alpha}$: $\mathbf{x}^T \boldsymbol{W} \boldsymbol{\alpha} + b$ where $\boldsymbol{\theta} = \boldsymbol{W} \in \mathbb{R}^{n \times k}, b \in \mathbb{R}$

This approach is called unsupervised since the dictionary Δ^* is obtained to find a sparse representation of N training time-series *independently* of the classification task. However, as pointed by Mairal et al., 2009, the dictionary found with this procedure is optimal in the sense of *reconstructive* tasks but not for *discriminative* one, that is, classification.

In order to tackle this issues, supervised dictionary learning has been introduced in order to learn a discriminative dictionary exploiting the class label information. In Mairal et al., 2009, the authors propose an approximation solution of formulation (2.9) which learn jointly Δ and $\boldsymbol{\theta}$:

$$\min_{\Delta, \boldsymbol{\theta}} \sum_{i=1}^{N} L(S^*(\mathbf{x}_i, \Delta, \boldsymbol{\theta}, -y_i) - S^*(\mathbf{x}_i, \Delta, \boldsymbol{\theta}, y_i)) + \lambda_2 \|\boldsymbol{\theta}\|_2^2 \tag{2.9}$$

where $S^*(\mathbf{x}_i, \Delta, \boldsymbol{\theta}, y_i) = \min_{\boldsymbol{\alpha}} L(y_i, f(\mathbf{x}_i, \boldsymbol{\alpha}_i, \Delta, \boldsymbol{\theta})) + \lambda_0 \|\mathbf{x}_i - \Delta \boldsymbol{\alpha}_i\|^2 + \lambda_1 \|\boldsymbol{\alpha}_i\|_1$ Then the classified label $\hat{y}$ of a new time series $\boldsymbol{x}_{new}$ is given by

$$\hat{y} = \arg\min_{y \in \{-1;+1\}} S^*(\mathbf{x}_{new}, \Delta, \boldsymbol{\theta}, y)$$

given the learned Δ and $\boldsymbol{\theta}$. Other supervised approaches are discriminative KSVD Zhang and Li, 2010, task-driven dictionary learning Mairal, Bach, and Ponce, 2012, Fisher discrimination dictionary learning Yang et al., 2011, and label-consistent KSVD (LC-KSVD) Jiang, Lin, and Davis, 2011; Jiang, Lin, and Davis, 2013. However, all these approaches are not robust to time-shifts or general deformations as it uses the Euclidean Distance. In order to overcome this issue, in Chen et al., 2015b a family of Gaussian elastic matching kernels was introduced. They use DTW, ERP, and TWED distances to compute the guassian kernel

$$K(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \exp -\frac{\|\mathbf{x}^{(1)} - \mathbf{x}^{(2)}\|^2}{2}.$$

However, the Gaussian elastic matching kernel cannot be guaranteed to be a positive definite symmetric (PDS) kernel. Thus, proper modifications have to be applied in order to remove the non-PSD part. Although the attempts to embed DTW distance to dictionary learning, several issues such as non-positive semi-definiteness of "DTW Gaussian kernel" can compromise the robustness of results. Moreover, this approach is computationally expensive.

Recently, in view of these considerations, another sparse representation approach of a dictionary has been considered. It relies on the well-known notion of *centroid* for clustering algorithms. Each centroid is considered as the class representative and thus, if the classification problem involves M classes, M representative time-series will be selected. However, DTW does not induce a proper definition of mean and thus the literature attempted to find a definition which is consistent with DTW. The most promising definition was given by Petitjean, Ketterlin, and Gançarski, 2011; Petitjean et al., 2016 and it was called DTW Barycenter Averaging (DBA). Roughly speaking, it is based on a expectation-maximization scheme and multiple sequence alignment (commonly used in computational biology). This method is very effective as it allows to apply numerosity reduction reducing N and DTW approximation to speed-up the single comparison. The difference between this approach and dictionary learning is that the "centroid" could not appertain to the dataset. Moreover, this method is supervised in some sense as it exploits the class label information by evaluating the centroid for each class.

Parametric distance-based methods

Parametric distance-based methods compute the distance onto a reduced parametric representation of the signal. In this case, each time-series $\mathbf{z}$ is represented by a representation $\mathbf{x} \in \mathbb{R}^p$.

The most common procedure for the training phase is as follows:

- Find a parametric representation of all the training time-series. The most used techniques are DWT, DFT stopped at a given coefficient order,
- Find a "centroid" or more generally a representative (template) of each class.

Once obtained the templates for each class, then the classification task is just given by a 1-NN classifier on the representatives $\mathbf{x}^{(\bullet)}$.

These simple methods suffer from from various issues as they do not care of the intrinsic "dynamics" information of the signal. This issue has been tackled, mainly, by the computer vision literature (Vishwanathan, Smola, and Vidal, 2007; Bissacco, Chiuso, and Soatto, 2007) and by Cuturi and Doucet, 2011; Chen et al., 2015a in a general context. Bissacco, Chiuso, and Soatto, 2007 proposes family of kernels for dynamical systems based on the Binet-Cauchy kernel Vishwanathan, Smola, et al., 2004 for recognizing dynamic textures. Bissacco, Chiuso, and Soatto, 2007 extends the work of Vishwanathan, Smola, and Vidal, 2007 considering phase information, inputs or initial conditions. Essentially, these two works rely on a probabilistic modeling of the time series to define a kernel: in order to compare two time-series, firstly, the dynamic behavior of each time-series is learned by learning the parameters of a given state space dynamical systems, and then, the kernel is defined as a kernel between these two sets of parameters. In other words, the distance is computed over the parameters of the fitted dynamical systems. We will see later, that a similar approach is followed by other classification methods. The work of Cuturi and Doucet, 2011 introduced the *Autoregressive Kernels* that are based on the vector autoregressive model (VAR): every multivariate (q-dimensional) time-series $\mathbf{z} \in \mathbb{R}^{q \times n}$ is represented by the feature $L(\theta; \mathbf{z}) = p_\theta(\mathbf{z})$, which is the likelihood function (it a function of θ for a fixed sample $\mathbf{z}$), modeled by a VAR model. Given a measurable space $\mathcal{X}$ and a model, i.e. a parametrized family of distribution on $\mathcal{X}$ of the form $\{p_\theta, \theta \in \Theta\}$, the kernel K of two time-series $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ is defined by

$$K(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = \int_{\theta \in \Theta} p_\theta(\mathbf{z}^{(1)}) p_\theta(\mathbf{z}^{(2)}) \omega(d\theta)$$

where, in this case, $\omega(d\theta)$ is the the matrix-normal inverse-Wishart prior. Moreover, Cuturi and Doucet, 2011 have shown that this kernel can be easily computed even when $q \gg n$ due to the fact that it does not resort to the actual estimation of a density. Indeed, all the kernels defined in Vishwanathan, Smola, and Vidal, 2007; Bissacco, Chiuso, and Soatto, 2007 and Cuturi and Doucet, 2011 rely on a probabilistic parametric modeling of time series, but the computation of the Autoregressive Kernel avoids the two step approach presented above. Finally, Chen et al., 2015a presents a model-metric co-learning (MMCL) methodology, which differently from the works on Vishwanathan, Smola, and Vidal, 2007; Bissacco, Chiuso, and Soatto, 2007; Cuturi and Doucet, 2011, present a kernel based on non-linear dynamical systems, named Echo State Networks (ESN) Jaeger, 2001; Jaeger, 2002. For each time series, an ESN-model is trained and the model parameters θ are used to compute an opportune distance, also using kernel methods. For other recent applications of ESN and its extension using Liquid State Machines (LSM), see Aswolinskiy, Reinhart, and Steil, 2016; Ma et al., 2016; Li, Hong, and Chen, 2016. We can notice that all the methods presented in this subsection can be seen as feature-based models. Indeed, in Cuturi and Doucet, 2011 the profile likelihood and in Vishwanathan, Smola, and Vidal, 2007; Bissacco, Chiuso, and Soatto, 2007; Chen et al., 2015a; Ma et al., 2016; Li, Hong, and Chen, 2016 the parameters of the dynamical systems, can be seen as features.

Although accounting the "dynamics" information using the kernel methods may lead to superior accuracy with respect to the simple parametric distance-based methods, they are more computationally expensive (as they requires the feature extraction phase and the computation of the kernel). This fact would favor the usage of simpler methods which avoid the computation of the kernel. Finally, in Table 2.3 a summary of the most important characteristics of DB methods is presented.

TABLE 2.3: Main characteristics of distance-based methods that have been reviewed in this work.

	Methods	**Characteristics**
Purely DB	DTW	Invariant to time-warpings Complexity is $O(n^2)$
	↳ DTW approximations	Complexity is $O(n)$
	↳ DDTW	DTW on the first-derivative signal Reduces pathological alignments
	↳ WDTW	Filtering with logistic weight function Favor matching points located in a neighborhood Reduces pathological alignments
	↳ ERP	Supports local time shifting Metric distance function
	↳TWED	Elastic distance measure Edit Distance + L_p norm
	↳ Alignment Kernels ↳ DTWUDC ↳ DTWUDC	DTW-based alignment kernel DTW + dynamic constraints DTW + dynamic constraints
Red-DB	SAX	Reduce into symbolic space
	↳ 1NN-SAX	SAX representation + SAX distance Suitable for simple signals α and p must be tuned Prediction is $O(N)$
	↳ 1NN-SAX With k templates	Prediction is $O(\tilde{M})$
	1NN-SFA	Fourier representation More adherence to the shape w.r.t. SAX Finer resolution with total re-computation
	Shapelets	Shift-invariant templates Training is $O(N^2n^3)$
	Dictionary Learning	Learn a sparse representation In general use euclidean distance Chen et al., 2015b adds kernel representation
	DBA	DTW centroids are defined and used as templates
Par-DB	1NN-DWT	Better than DFT to handle non-stationarity Good frequency and temporal resolution
	Binet-Cauchy Kernels	Kernel to embed dynamic behavior Rely on a probabilistic parametric modeling of time series
	Autoregressive Kernels	Easily computed even when $q \gg n$ Rely on a probabilistic parametric modeling of time series AR model
	MMCL	Kernel based on non-linear dynamical systems (ESN)
	LSM	Distance on LSM parameters

Chapter 3

Activity recognition using Sparse representations

3.1 AR procedures

The following section serves as an introduction to both Chapter 3 and Chapter 4.

Activity Recognition (AR) is a prominent research area with applications to home automation (Belgioioso et al., 2014), gaming (Gowing et al., 2014), sport (Cenedese, Susto, and Terzi, 2016) and health care (Clifton et al., 2013). In particular, the rapid growth of IMUs (Inertial-Measurement Units) has allowed, in recent years, the development of compact sensor-equipped devices (e.g. smartwatches and smartphones), which led efficient monitoring of human activities to be feasible and to have a strong impact on the quality of life (Clifton et al., 2013).

On the other hand, wearable devices present some limitations in terms of computational capability and memory, which force the algorithm design to be at the same time efficient and simple. Moreover, decision algorithms need to be portable, i.e., classification tasks have to be taken at a device-level (Cenedese et al., 2015).

It is important in this context to differentiate between *activities* and *gestures*. Gestures (also called in the following *atomic gestures*) are here considered as basis movements that compose an activity that is conversely completely characterized by one or more atomic gestures: for example, in swimming, a stroke (gesture) completely characterizes the style (activity).

Due to the vastness of application scenarios, it is helpful to categorize the AR problems into three main types:

- *continuous-repetitive* - activities that are continuous and composed by repeated gestures with a periodic behaviour within the same activity type. E.g.: swimming, running;
- *continuous-spot* - continuous activities with non-repetitive gestures. E.g. tennis playing, and daily activities;
- *isolated* - activities composed by isolated gestures: the data stream includes only an activity period and it is exactly known when the activity starts and when it stops. E.g.: command gestures.

The main challenge in applying ML algorithms in AR problems is to translate the informative content contained in the IMU-generated time series into a static format that can be handled by ML classifiers (G.A. et al., 2016; Cenedese, Susto, and Terzi, 2016). Typically, this is achieved with a flow chart of operations as in the scheme depicted in Fig. 3.1: the pipeline contains two blocks, window extraction and feature extraction, aiming to translate the informative content in the classical form $X \in \mathbb{R}^{N \times p}$, where N and p are, respectively, the number of observations (time windows in this context) and the number of features extracted from each window. This approach will be adopted in Chapter 3.

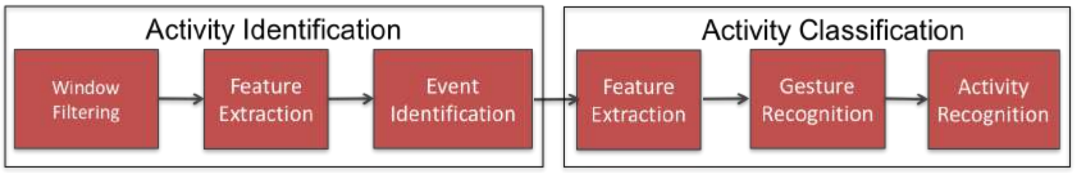


FIGURE 3.1: Scheme of the classical ML approach to AR/GR problems.

The main drawback of using window-based methods is that there is no one-to-one correspondence between gestures and windows; in fact, gestures could have different shapes (at least locally) and warping in time and amplitude domains that dramatically change gestures duration and the window-based statistics (features) $\phi \in \mathbb{R}^p$. Moreover, the aforementioned approach has other two major issues: (i) single gestures are not isolated, which of course make the approach not feasible for GR problems; (ii) the feature extraction phase may be computationally expensive, causing this approach to be almost impracticable for wearable applications.

An alternative procedure suggests to directly compare raw signals with specific distance definitions and with the usage of a distance-based classifier (like Nearest Neighbor, NN (Susto et al., 2015)); classifying directly on the time-series allows to bypass the feature extraction phase, but the window extraction procedure is still required. In this sense, one of the most popular approaches for defining a distance between time series of different length is the Dynamic Time Warping (DTW) (Berndt and Clifford, 1994)): unfortunately, DTW is a dynamic programming technique that hardly meets computational complexity and memory requirements of wearable devices. In Chapter 4 we will use a symbolic distance-based approach.

3.2 Contribution

As we introduced in Chapter 1, the technological advances have led to the development of a generation of wearable devices, more suitable for a daily-life user experience in a wide range of application from biomedical studies to work and leisure activities (see e.g. Mathie et al., 2004; Albinali et al., 2010).

Along the methodological line that adopts ML techniques, the contribution of this chapter is concerned with the design of a procedure that is able to detect if a gesture occurs (Event Identification phase), to characterize it with a specific signature (Feature Extraction phase), and finally to classify a newly measured gesture according to the class dictionary (Gesture Recognition phase). The specialized literature reports different ML and pattern recognition techniques to classify a gesture from the captured data, from simple models such as k-Nearest Neighbor to more complex ones as Hidden Markov Models and Support Vector Machines (SVMs) (e.g. (Liu, Pan, and Li, 2010; Wu et al., 2009)). In such a context, the SVMs methods result to be the most effective but may suffer from not being parsimonious. The specific flavor is posed on the parsimony of the solution, and in this respect we develop a powerful framework based on Relevant Vector Machines (RVMs) Tipping, 2001, which is able to cope with gesture complexity but at the same time is designed to be sparse and memory-effective, and therefore capable of meeting the resource constraints of wearables (much more limited than smartphones).

Furthermore, as a case study, this work discusses the application to cross-country skiing, in particular the classic technique. The most recent works specifically related to this subject are Holst and Jonasson, 2013, Stöggl et al., 2014 and Fasel et al., 2015. In particular, Holst and Jonasson, 2013 and Stöggl et al., 2014 deal with cross-country skating using a smartphone placed on the user chest and adopt a Gaussian Markov model to characterize gestures while Fasel et al., 2015 studies spatio-temporal parameters to describe the gesture style. We face new raising issues w.r.t these contributions, namely:

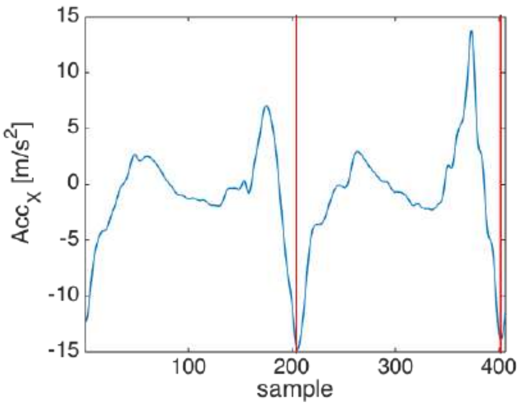
- *device orientation*. A fixed and "stable" position for sensors is employed in such literature, meanwhile in our case sensors are placed on the wrist of the user, resulting in an increasing number of degrees of freedom.
- *resource constraint*. Algorithm complexity must be bound in order to run on a smart-watch or similar device in a quasi real-time fashion.
- *dataset heterogeneity*. Differently from the given references, where experiments are performed in a controlled environment and all users are athletes, in this chapter several skill levels and a heterogeneous set of users are considered.
- *application data and data model*. Holst and Jonasson, 2013-Stöggl et al., 2014 study skating techniques while here classic techniques are in focus. Finally, Stöggl et al., 2014 refers to atomic gestures recognition while pattern of gestures are sought in the approach described here.

The remainder of the chapter is organized as follows: Sec. 3.3 addresses the pre-processing and orientation-dependency issues, Sec. 3.4-Sec. 3.5 introduce the ML classification problem and the adopted RVM-framework, in particular defining the Event Identification and Gesture Recognition phases. Hence, Sec. 3.6 reports the experimental validation on the case study and provides a discussion over the obtained results. Finally, Sec. 3.7 summarizes the conclusions of the work outlining current and future research avenues.

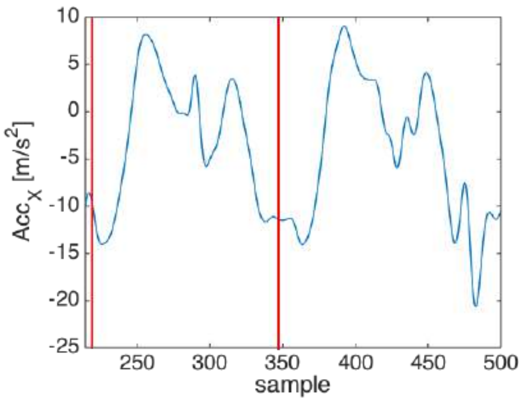
3.3 Data Pre-processing

Given the data-driven approach exploited by the ML procedures, a crucial issue arising in this sort of application concerns the variance of the input signals. In our specific context, the sensory platform is placed on the user wrist, and the measurement data are 3-D accelerations: in this context, noise and orientation variability play a crucial role on gesture classification. Indeed, the appearance of gestures (e.g. the gesture shapes) are orientation-dependent, user-dependent and prone to various sources of nuisance, such as measurement noise, bad-performed gestures, and so on. For example, if two athletes perform the same gesture but with two different wrist orientations, the acceleration signals are totally different. Fig. 3.2 provides an example of gesture variability for Cross-country skiing in the classic styles, which include Double Poling (DP), Kick Double Poling (KDP) and Diagonal Stride (DS). Thus, we would need a orientation normalization in order to compensate for these (unwanted) degrees of freedom.

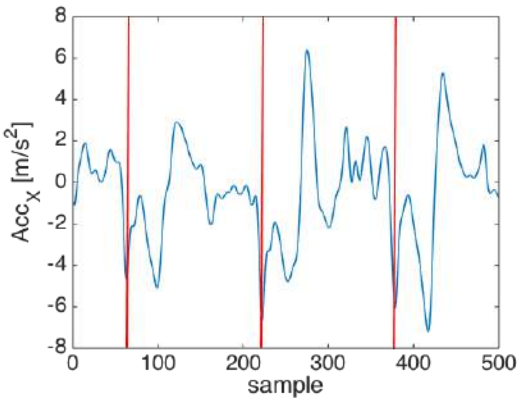
In this work, gesture recognition is an algorithm based on 3-D accelerations. Unfortunately, accelerometers don't permit to infer the linear accelerations (i.e. accelerations due to human motion) since the device orientation is not known. Also, given the heterogeneity of the employed dataset, different wrist or device orientations may



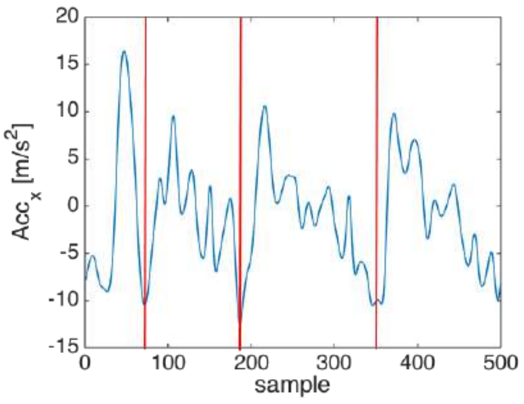
(A) DP$_1$



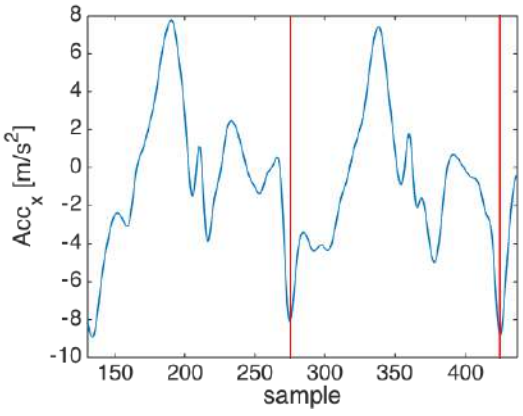
(B) DP$_2$



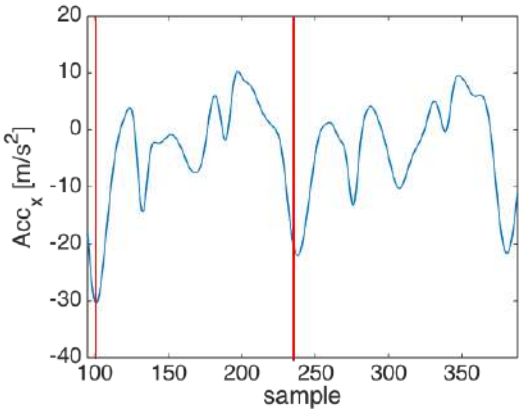
(C) KDP$_1$



(D) KDP$_2$



(E) DS$_1$



(F) DS$_2$

FIGURE 3.2: Sample acceleration data (X-axis only) for different gesture types: DP, KDP, DS. Each row refers to an action performed by two different users. In this plot, the gesture instances are manually identified (vertical red lines) to highlight the high variability in signal amplitude and duration both between different users and within the same user.

in principle affect the measurements and require a personalized calibration of the device.[1] To partially overcome these issues, a sensor fusion Madgwick, 2010 approach is adopted (by composing all the information coming from a 9-axis MEMS IMU) to retrieve the linear accelerations.

Conversely, when the sensors are subject to other additional forces, the fusion filter cannot exactly infer such forces and a wrong estimation of the accelerations is obtained. Nevertheless, with the problem at hand, a careful setting of filter gain can effectively reduce biases and other relative-orientation issues: thus, in this work, we consider accelerations $\mathbf{a}^E = \left(a_x^E, a_y^E, a_z^E\right)$ w.r.t. the Earth's frame.

Then, after a Gaussian filtering step to smooth signals, a Principal Component Analysis (PCA) is performed in order to verify which axis carries more information and indeed it can be oserved that in the device reference frame, almost all the information is related to the X-axis, which follows the direction of user's arm, while in the world reference frame, the situation is less clear and also Y-axis and Z-axis carry useful information.

We note that the pre-processing phase employs standard tools generally available in embedded system libraries, and is then perfectly suitable for a real-time use for classification.

3.4 ML Classification Problem

Within the range of applications of ML techniques, the case of gesture/activity recognition may be actually seen as a *supervised classification problem* (see also Bishop et al., 2006; Hastie, Tibshirani, and Friedman, 2009; G.A. et al., 2016).

Formally, a training dataset $\mathcal{T}$ is considered

$$\mathcal{T} = \{x_i \in \mathbb{R}^{1\times p}, y_i \in \mathbb{D}\}_{i=1}^{N}, \tag{3.1}$$

where N labeled observations are available: the i-th observation is characterized by p input features and a scalar target value y_i (*label*) that belongs to a certain domain $\mathbb{D}$ representing the set of classes. Then, for each newly acquired observation x, a function $y(x)$ is computed and x is classified according to $\mathcal{T}$ and the defined classes. In the specific application to gesture recognition, the input $x \in \mathbb{R}^{1\times p}$ is obtained from the 3D acceleration time series $\mathbf{a}^E(t) \in \mathbb{R}^{3\times N}$, $t = 1, \ldots, N$, by defining p features that characterize the gesture using feature extraction, while y belongs to the space of the known gestures.

As already mentioned, SVMs are probably the most popular ML approach to classification thanks to the performance they yield and the availability of optimized algorithms for their computation Susto et al., 2015. In this framework, the classification function of a binary classifier is expressed as

$$z(x) = \sum_{i=1}^{N} w_i \phi_i(x), \tag{3.2}$$

[1]Differently from the home automation application where the device is designed to serve several people, in this case the system is personal and therefore a user specific calibration can be indeed implemented.

where w_i are weights and $\phi_i(x)$ are *basis functions*; a popular choice of the $\phi_i(x)$ is the Radial Basis Function (RBF) $\phi_i(x) = K_{RBF}(x, x_i)$

$$K_{RBF}(x, x_i) = e^{-\gamma \|x - x_i\|^2}, \tag{3.3}$$

where $\gamma \in \mathbb{R}_+$. A multiclass classification can be easily obtained by one-vs-all technique or other voting strategy.

Finally, the performance measures that are generally employed in the cross-validation (CV) are the classification accuracy, namely the percentage of correctly classified samples, and the confusion matrix, which details how misclassified samples are distributed among the classes.

3.5 RVM-based Framework

Activity Recognition problems, like the one under investigation, require a huge number of training examples in order to accurately describe the gestures to be recognized. Unfortunately, the number of support vectors that describes the binary decision typically grows linearly with the size of the training, resulting in too many support vectors to be stored and thus raising a critical issue with the use of SVMs for dealing with this specific task from a computational cost point of view. In actual fact, another drawback of SVMs is related to the deterministic form of the solution that gives no indications regarding the confidence level of the classification.

To overcome these issues, Relevant Vector Machines (RVMs) (Tipping, 2001; Zanon, Susto, and McLoone, 2014) are here adopted, which employ an identical functional form to the SVMs (Belgioioso et al., 2014) but within a sparse probabilistic framework. As a drawback for adopting RVMs over SVMs, the training phase is generally slower.

With such premises, in the following we present a classification algorithm based on Sparse Bayesian Learning (SBL) and RVMs, which aims at providing parsimonious solutions to meet the requirements of the application in exam. The candidate functional for the models is expressed as in (3.2).

After the pre-processing phase described before, signal $\mathbf{a}^E = \left(a_x^E, a_y^E, a_z^E\right)$ is sequentially divided in frames of length L_w with an overlap of length L_o. Suitable choices for values L_w, L_o depend on the specific data at hand: the sensitivity of the proposed algorithm to these parameters will be discussed in the remaining part of this section. Each frame is given as input to the algorithm, which consists of two main phases, namely:

1. Event Identification (EI)
2. Gesture Classification (GR)

where a step of feature extraction is specifically performed for both EI and GR.

The aim of the procedure is the identification of a model $\widehat{\mathcal{M}}_{EI}$ able to detect a gesture (during EI), and a model $\widehat{\mathcal{M}}_{GR}$ for classifying novel gesture instances (during GR). The entire model is referred as $\widehat{\mathcal{M}} = \widehat{\mathcal{M}}_{GR} \circ \widehat{\mathcal{M}}_{EI}$. In our approach (3.2) becomes

$$z_f(x) = \sum_{i=1}^{N} w_i^f e^{-\gamma_f \|x - x_i\|^2} + w_0^f \tag{3.4}$$

where f is either EI or GR and for $\widehat{\mathcal{M}}_{EI}$ and $\widehat{\mathcal{M}}_{GR}$ respectively.

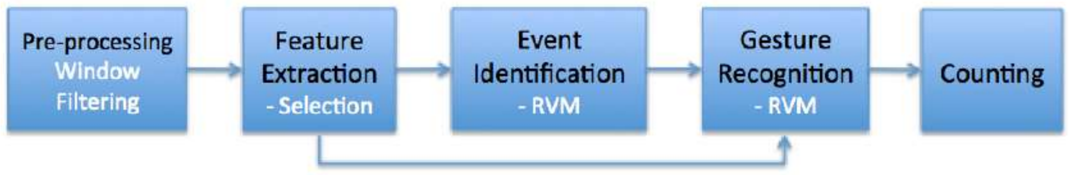


FIGURE 3.3: Flowchart of the whole procedure for gesture classification.

Considering the EI phase, we refer to $\mathcal{M}_{EI}^{i}$ as the i-th model built at step i (with i features), MCR_{EI} as the Misclassification Error (%), τ_{TOL} as the tolerance factor, $\hat{y}_{EI}^{i}$ as the $i-$th predicted label; we use the same notation for GR. MCR represents the total misclassification error, expressed in %. A flowchart of the whole algorithm is shown in Fig. 3.3: Sec. 3.5.1 details feature extraction, while Sec. 3.5.2 details EI and GR phases; also, the pseudo-code of whole procedure is given in Algorithm 1 and Algorithm 2.

Algorithm 1 Gesture recognition algorithm: Training

```
Input: $\mathcal{T}^{train}$
Output: $\widehat{\mathcal{M}}$
1: Gaussian filtering
2: Frames of $L_w$ and $L_o$ overlap
       - Determine $L_w$ and $L_o$ by CV.
       - The number of training segments is $N^{train} = \frac{N-L_o}{L_w-L_o}$.
       - Build the training set $\mathcal{S}^{train}$ of frames.
3: Extract features $F \in \mathbb{R}^{1\times p}$ and build the design matrix $D \in \mathbb{R}^{N\times p}$ from $\mathcal{S}^{train}$.
4: Select informative features $F_1 \subset F$ by the t-test ($\alpha = 5\%$)
5: Sequential Forward Feature Selection
   $F_2 \leftarrow \varnothing, i = 1$
   while MCR$_{i-1}$ - MCR$_i > \tau_{\text{TOL}}$ do
   $F_2 \leftarrow F_2 + F_1^i$                                  ▷ Add $i-$th feature
   Train RVM model $\mathcal{M}_{EI}^{i}$ on $\mathcal{S}^{train}$
   Compute MCR$_i$ by a $5 \times 2$ CV
   end while
   return $F_2 \subset F_1 \subset F$
7: Select the hyper-parameter $\gamma_{EI}$ by CV on $\mathcal{S}^{train}$
8: Train the final model $\widehat{\mathcal{M}}_{EI}$
9: Repeat from Step 4 for Gesture Classification (only for true gestures)
```

3.5.1 Feature Extraction

For both EI and GR a common preliminary feature extraction procedure is run in order to learn the model $\widehat{\mathcal{M}}$. For this purpose, 9 clusters (groups) of features are computed from the acceleration signal $\mathbf{a}^E$ (for each axis):

- α: autocorrelation-based features. Repetitive gestures present a peak in the autocorrelation function at the frequency $1/\tau$, where τ is the peak lag;
- $s^2/\mu/r$: sample variance/mean/RMS value;

Algorithm 2 Gesture recognition algorithm: Test scenario

```
Input: $\widehat{\mathcal{M}}_{EI}, \widehat{\mathcal{M}}_{GR}, \mathcal{T}^{test}$
Output: $\hat{y}_{GR}, MCR$
 1: Do gaussian filtering and segmentation as in Algorithm 1.
 2: Extract informative features $F_{EI}$ and build the design matrix $D_{EI}$ from $\mathcal{S}^{test}$
        $F_{EI} \in \mathbb{R}^{1\times 16},\ D_{EI} \in \mathbb{R}^{N\times 16}$
 3: $\hat{y}_{EI} \leftarrow$ predict $\left(D_{EI}, \widehat{\mathcal{M}}_{EI}\right)$
 4: Evaluate $MCR_{EI}$
 5: Let $\hat{\mathcal{S}}_{EI}^{test}$ the $N_{EI}$ test samples (frames) predicted as gestures.
 6: Extract informative features $F_{GR}$ and build the design matrix $D_{GR}$ from $\hat{\mathcal{S}}_{EI}^{test}$
        $F_{GR} \in \mathbb{R}^{1\times 21},\ D_{GR} \in \mathbb{R}^{N_{EI}\times 21}$
 7: $\hat{y}_{GR} \leftarrow$ predict $\left(D_{GR}, \widehat{\mathcal{M}}_{GR}\right)$
 8: Output filtering
    if $\hat{y}_{GR}^{i-1} = \hat{y}_{GR}^{i+1}$ & $\hat{y}_{GR}^{i} =$ DS then
        $\hat{y}_{GR}^{i} \leftarrow \hat{y}_{GR}^{i+1}$
    end if
11: Evaluate $MCR_{GR}$ and $MCR = MCR_{EI} \cdot MCR_{GR}$
```

- ε: energy value as $\varepsilon = \frac{1}{L}\sum_{i=t}^{t+\tau} |d_i|^2$, where $\{d_i\}$ are the coefficients of DFT, t and τ are employed to select the band of interest;
- P: power spectrum band $\{P_i\}_{i=1,\dots,15} = \frac{1}{L}|d_i|^2$, computed at 15 signal frequencies (up to $3.3Hz$);
- a_{min}/a_{max}: minimum and maximum value;
- ρ: axis correlation value.

In particular, 14 features are extracted from the autocorrelation function:

- number of peaks;
- number of prominent peaks, i.e., peaks higher than a threshold T_p and higher than their nearest neighbors;
- number of peaks higher than a threshold T_p;
- location/peak value/width/area/order/height of two highest peaks.

The features are grouped in the vector F:

$$F = \left[\alpha\, s^2\, \mu\, r\, \varepsilon\, P\, a_{min}\, a_{max}\, \rho\right] \in \mathbb{R}^{1\times p}, \quad p = 108$$

In the problem at hand, we look for a compact solution, both in computational terms and in memory occupancy. Hence, it is essential to remove from F all the less informative and redundant features. To this aim a two-fold procedure is conducted (separately for EI and GR):

1. two-sample t-test in order to capture only informative features given by the p-value;
2. sequential feature selection to remove redundancy.

The t-test assumes all features to be independent and thus it is not able to remove redundancy due to correlated features. Hence, a forward sequential selection is performed. It is important to note that a cross-validation procedure is needed in order to obtain an unbiased estimation and avoid over-fitting. In our case, a 5x2 CV is employed (Dietterich, 1998).

3.5.2 Event Identification and Gesture Recognition

EI aims at recognizing whether a gesture occurs. An event is identified as a gesture through an hard binary decision (0-1) provided by a RVM classifier $\widehat{\mathcal{M}}_{EI}$ with the most informative *predictors*. Each frame is classified as either *non-gesture (NG)* or *gesture (G)*; in this setting 'transition frames' containing both gesture and non-gesture observations are hardly classified as *non-gesture* or *gesture*. Nevertheless classification is based on the output of an estimate of the probability of belonging to classes 0 and 1 (Bayesian approach); hence, transitions may be recognized by analyzing the probability trend over overlapping frames.

After a gesture is recognized, the task is to identify which class c_i the gesture belongs to (GR phase). In general we consider a n-ary classification problem. For convenience, we indicate the class set as $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$. As in the EI phase, a set of good predictors is selected in order to build the model $\widehat{\mathcal{M}}_{GR}$ employed to classify classes in $\mathcal{C}$.

In order to improve the robustness of the algorithm, a median filtering on the output sequence is applied. In addition to low complexity, the proposed framework has the advantage of the probabilistic approach; in fact, SBL provides an effective (differently from SVM) estimate of likelihood that a certain observation belongs to a class c_i. In our case, this means that in the EI phase the algorithm is able to handle transitions. In fact, let's suppose a transition is occurring, i.e. there is a frame containing both gesture and non-gesture samples; say, l_g the number of samples representing a gesture and $l_w - l_g$ samples representing a non-gesture; then the output probability is $P(y = +1 \mid x_i) = \frac{l_g}{l_w}$.

3.6 Experimental Validation

In this work, acceleration data have been obtained from 8 users for a total of 41 datasets, performing gestures in the classes DP, KDP, DS. We will refer to non-gestures as AOM, which are present given the real world conditions of the experiments. In detail, users have been asked to perform sessions of a single gesture, i.e. transitions between different gestures are not considered in this experiment, while transitions between gesture and non-gesture are accounted for (an example of the data is reported in Fig. 3.2).

3.6.1 Tuning and Cross-Validation

Generally, ML algorithms require a phase of parameter tuning upon which the model accuracy and the classification algorithm performance strongly depend. In particular, in the case of continuous values, such as the kernel scale parameter γ, a *MCCV cross-validation* procedure has been employed.

Table 3.1 summarizes the parameters to be tuned, for each step of the procedure, and their respective values.

TABLE 3.1: Cross-validated parameters.

Phase	DoF
Pre-processing	$\sigma = 0.4\,s$ (Gaussian filter)
Event Identification	$L_w = 5\,s$, $L_o = 0.48\,s, \gamma = 0.18$
Classification	$\gamma = 0.22$ (RBF kernel)

As we can see in Table 3.1, the window length is $L_w = 5\,s$. This parameter has a strong dependency on the frequencies of gestures and thus, a cross-validation is needed for different applications. In the specific problem, fundamental frequencies correspond to an interval of $1 - 2\,s$, and thus $L_w = 5\,s$ is a reasonable solution allowing to capturing enough occurrencies. Moreover, in order to handle transitions we chose $L_o = 480\,ms$. This doesn't imply that such an overlap has to be employed in the training phase. Empirically, we found that with overlap of $L_w/2$ the model $\mathcal{M}$ *captures* enough information to explain the data. As a matter of fact, setting L_o to an high value in the training phase only causes an increased number of Support Vectors, which not necessarily yields an improved accuracy. This behavior can be explained by observing that what really matters is the flexibility of the model, namely that the model should be able to treat all the possible configurations of patterns.

3.6.2 Classification Results and Performances

In the remainder of this section we present the results of the style recognition experiment on the Cross-country skiing data. Models have been trained on a training set of 70% of the whole dataset and tested in the remaining 30%. First we proceed to present results from feature extraction phase and then from classification in EI and GR phases.

The t-test (both for EI and GR) indicates that 90% of features are informative and thus only 10% of features are discarded. This is due to the fact that features are highly correlated and, indeed, forward sequential feature selection procedure is employed to remove redundancy. In the EI and GR phases, 16 and 21 independent features are detected, respectively. In the EI phase, autocorrelation features play an essential role since they characterize repetitive gestures while in GR, spectra, variance and correlation features are key to capture different gesture signatures.

The results of the classification in term of accuracy and confusion matrix are reported in Table 3.2 and Table 3.3.

TABLE 3.2: Results of the classification with classes DP, KDP, DS.

		Predicted		
		DP	KDP	DS
True	DP	662 (92.7%)	99 (25.8%)	44 (13.1%)
	KDP	46 (6.4%)	265 (69%)	33 (9.9%)
	DS	6 (0.84%)	20 (5.2%)	257 (76.9%)

TABLE 3.3: RVM results with DP and KDP in a single class.

		Predicted				Predicted	
		NG	G			(K)DP	DS
True	NG	245 (94.9%)	13 (0.3%)	True	(K)DP	317 (93.5%)	22 (5.3%)
	G	4 (5.1%)	1419 (99.7%)		DS	17 (6.5%)	312 (94.7%)

In the EI phase, the algorithm fails only in situations similar to that reported in Fig. 3.4, i.e. where the gesture is not well-defined: a filtering action on the output partially solves this problem but unfortunately it is limited by the window length, i.e. it is not allowed to have a sub-segment resolution.

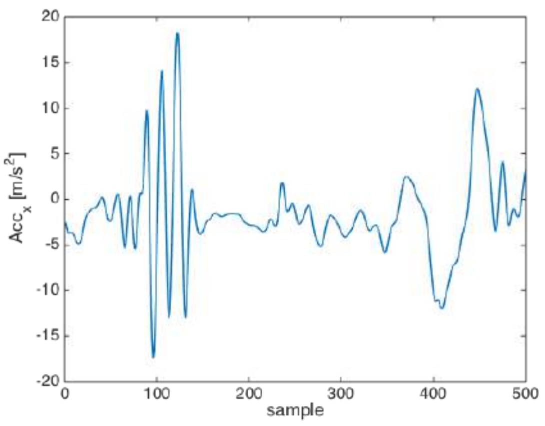


FIGURE 3.4: Acceleration data related to the X-axis (w.r.t. the Earth's frame) when a gesture is performed. In this case, the algorithm is not able to capture the periodicity and the other relevant features.

The GR phase presents a more critical issue: DP and KDP gestures only differ by a kick; as a consequence, given the sensor position, it is very difficult to capture leg accelerations.

For this reason, we decided to employ an hierarchical classification: run EI phase as above, then run the classification over classes DP+KDP (i.e. the union of DP and KDP) and DS; finally classify DP and KDP with a third RVM classifier.

Unfortunately, the results are not satisfying as reported in Table 3.2. A different position of sensor may probably solve this problem. Due to this fact, we merge DP and KDP in a unique class (K)DP, and the general results are presented in Table 3.3[2]. As we can see, the recognition performance remarkably improves.

Beyond the classification ability, one essential fact for the implementation on embedded systems tailored for sport applications is the low complexity of the resulting model: this target can be attained with the proposed sparse probabilistic approach as demonstrated in Table 3.4, where a comparison with SVMs has been carried out. Remarkably, RVMs provides a more sparse solution than SVMs by a factor 80.

Furthermore, another main feature of the RVM framework that has been mentioned is the capability of handling gesture transitions by looking at the probability $P(y = c_i \mid x_i)$.

[2]W.r.t. Table 3.3, it may be noticed that the number of true gestures detected by the algorithm in the EI phase is not equal to the total number of gestures in the table on the right. This is because class DP+KDP and DS are approximately equiprobable; so, we selected balanced random sets of each class.

TABLE 3.4: Comparison between RVM and SVM solutions in terms of complexity and performance.

	Complexity		**Accuracy (%)**	
	RVs	SVs	RVM	SVM
EI	8	512	98.9	97
GR	29	2545	94.7	92.7
Total	37	3057	93	90.5

			USER#1	**USER#2**	**USER#3**	**USER#4**	**USER#5**	**USER#6**	**USER#7**
SVM	EI	[SVs]	515	536	532	544	517	520	470
		[Acc %]	98.5	99.3	99.3	99.6	99.6	98.8	98.2
	GR	[SVs]	2688	2308	2970	2879	2758	2792	2732
		[Acc %]	91.3	84.2	97	82.2	93	93.5	95
RVM	EI	[SVs]	9	11	9	7	7	9	10
		[Acc %]	99.1	99.6	100	100	99.	98.8	97.9
	GR	[SVs]	148	102	156	28	154	136	62
		[Acc %]	96.8	97.5	95.7	93.6	95.6	96	94.5

TABLE 3.5: General performance in terms of accuracy and sparsity on EI and GR. Models are built on $K - 1$ users and tested on the K-th.

3.6.3 Discussion

As a final remark, the role of users on gesture realization is here addressed. More properly, we want to analyze the predictability of different user's gesture given the past data. In other words, we are interested in understanding if we are allowed to neglect user-dependency at the expense of a slight decrease of performance. In order to understand this aspect of the problem, we proceed as follows: a model from $K - 1$ users is trained and then we predict over the K-th user; this is repeated for all the users. We run this test only over 7 users since one executed only DP style. As we can see in Table 3.5 in EI phase all users, have accuracy near to 100% while in GR the forth user's gestures are not well-predicted by other users. This is confirmed by the number of RVs, which, without user 4 are remarkably fewer indicating that this user carries new information.

Indeed, the main issue we faced in this work is the dataset composition: all gestures tend to lose their own signature w.r.t. the variations of environment conditions, such as user level and wrist orientation. In addition, acquisitions didn't provide a golden reference (e.g. a video-reference) to be compared with the device data. In such a context, a crucial aspect of these kind of problem is indeed the boundary within which a gesture c_i is considered a "true" c_i. For example, the acceleration signal in Fig. 3.4 does not apparently represent a gesture even though it may be labeled as such.

3.7 Comments

We would like to remark that the algorithm has been tested on an heterogeneous XC-skiing dataset with users of different levels, in a real session environment with orientation changes and no constraint on gesture execution; in such a scenario, the RVM-based classification approach reveals to be not only less computationally expensive than SVM, but also more accurate.

Moreover, although the reported experiments are related on a single sport (XC-skiing), the presented framework has general purpose and could be employed for sports that present repetitive and quasi-repetitive actions. In this sense, the adaptation and performance assessment of the proposed methodology in new sport applications is currently under investigation.

Chapter 4

A symbolic approach to activity recognition

This chapter is focused on the *continuous-repetitive* type of activity (Morris et al., 2014), that is typical of sports (e.g. rowing and swimming) and health monitoring applications. In the following, we will also refer to a gesture as *atomic gestures*.

AR problems are usually solved by means of Machine Learning (ML) approaches, however, the aforementioned restrictions on computation and memory capacity cause great limitations in choosing the ML algorithms to be employed. For instance, in (Cenedese, Susto, and Terzi, 2016), Relevance Vector Machines are chosen over popular Support Vector Machines in order to meet the parsimony constraint required by wearables in the ML algorithm complexity.

Within this context the above restrictions can otherwise be handled by adopting symbolic representation techniques (Rajagopalan and Ray, 2006) in the treatment of IMU-generated time series data; with symbolic approaches, time-series are mapped into *strings*, which implies dimensional and numerosity reduction. Moreover, symbolic representations allow avoiding one pre-processing phase, called *Feature Extraction*, which is common to AR solution and often critical in the selection of parameters to be retained in the models.

With these premises, the contributions of this chapter are:

1. a symbolic approach based on Symbolic Aggregate ApproXimation (SAX) (Lin et al., 2003b) for AR. SAX is a popular symbolic approach intended for univariate time-series: since in many AR problems multiple IMU-generated time-series are available, we extend here the approach to multi-dimensional time series in order to exploit mutual information from multiple axes; the results of our experiments confirm that this multi-dimensional extension leads to superior accuracy w.r.t. univariate approaches.

2. a procedure to extract atomic gestures and a classification model for Gesture Recognition (GR) is built directly on gestures; more interestingly, a model that is *invariant* to duration and amplitude warpings is depicted. Then, we perform AR starting from GR classification results, through a window-based approach;

3. an *Event Identification* (EI) procedure is designed in order to detect time windows where activities to be identified are not performed; these time windows are labelled as *all the other movements* (AOM).

The remainder of the chapter is organized as follows: Sec. 4.1 is dedicated to discuss the approach and mathematical tools. Sec. 4.2 univariate symbolic classification problem is illustrated; Sec. 4.3 discusses the multivariate extension of the SAX approach while the gesture extraction phase is depicted in Sec. 4.4 and in Sec. 4.5 the experimental results are shown.

4.1 Symbolic AR procedure

As stated above, AR/GR problems are usually tackled by means of ML approaches (Morris et al., 2014); more precisely AR/GR problems are generally *classification* ones: the activity or gesture in exam has to be associated with one of the a-priori defined K possible classes of activities/gestures $\mathcal{C} = \{c_i\}_{i=1}^{K}$.

In this chapter, we adopt a symbolic approach, which is graphically summarized in Fig. 4.1. The proposed method is not relying on window-based extraction proce-

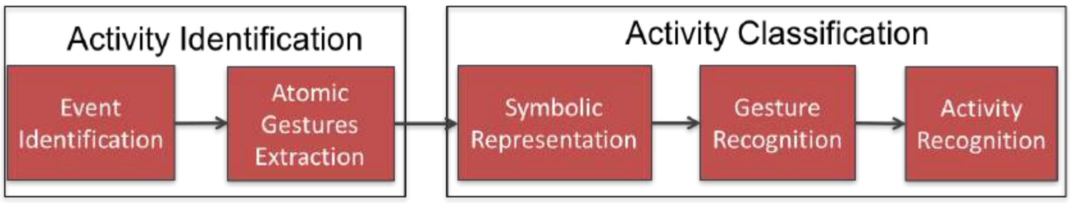


FIGURE 4.1: Scheme of the symbolic approach to AR/GR problems adopted in this work.

dures, but exploits a preliminary *atomic gesture* extraction phase. After the extraction phase, each gesture is then symbolized trough the SAX technique (briefly described in the following); afterwards, a classification model is built over the symbolic representation of a input collection $\mathcal{X} = \{X_i \in \mathbb{R}^{m \times p}\}_{i=1}^{N}$, where m is the dimensionality, p the feature cardinality and N represents the number of gestures. The *Activity Classification* phase will be detailed in Sec. 4.2, Sec. 4.3 then in Sec. 4.4 the *Activity Identification* phase is described.

We recall the SAX technique presented in Sec. 2.2.2 mainly consists on 3 phases:

- signals standardization in order to obtain a zero mean and unit variance signal;
- Piecewise Aggregate Approximation (PAA) (Keogh et al., 2001), described in the following;
- symbolic mapping through discretization on amplitude domain.

After normalization, in the PAA phase, a signal $\mathbf{z} = z_1, \ldots, z_n$ let $\bar{\mathbf{z}} = \bar{z}_1, \ldots, \bar{z}_p$ of length s is discretized on time in p frames in order to obtain a vector $\bar{\mathbf{z}} = \bar{z}_1, \ldots, \bar{z}_p \in \mathbb{R}^p$. Formally, the resulting i-th element $\bar{z}_i$ is defined by the mean of i-th interval:

$$\bar{z}_i = \frac{p}{s} \sum_{j=\frac{s}{p}(i-1)+1}^{\frac{s}{p}i} z_j \tag{4.1}$$

Then, the SAX representation procedure (i.e. the discretization on amplitude domain) can be summarized as follows. Let a_i denote the i-th element of the alphabet $\mathcal{A}$, with $|\mathcal{A}| = \alpha$. The mapping from the PAA approximation to the correspondent word $\mathbf{x} = x_1, \ldots, x_p$ of length p is obtained as follow:

$$x_i = a_j \qquad \text{iif} \qquad \beta_{j-1} \leqslant \bar{z}_i < \beta_j, \tag{4.2}$$

where $\{\beta_j\}_{j=1}^{\alpha-1}$ are break-points tuned to have symbols with equiprobable occurrence. One of the advantages of introducing the *SAX representation*, is that a new distance measure - which is a lower bound of euclidean distance - can be immediately defined. Let $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ be two time-series of same length n and $\mathbf{x}^{(1)} = x_1^{(1)}, \ldots, x_p^{(1)}$ and

$\mathbf{x}^{(2)} = x_1^{(2)}, \dots, x_p^{(2)}$ be their *SAX symbolic representation*; the SAX distance is defined as:

$$D_{SAX}(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = \sqrt{\frac{n}{p} \sum_{i=1}^{p} dist\left(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}\right)^2}. \tag{4.3}$$

4.2 Activity Classification for 1-D signals

To tackle the AR problem, it is convenient to have gestures that are normalized in duration; for this reason we assume in the following that fixed length gestures are available, without loss of generalization since resampling procedures can be in place before the Activity Classification phase.

Let assume of having N 1-dimensional gestures (or more generally signals) $g_i \in \mathbb{R}^{1\times q}$ of *fixed* length q, collected in a row-wise matrix $X = [g_1; \cdots; g_N] \in \mathbb{R}^{N\times q}$; a classification problem with K classes of gestures/activities $\mathcal{C} = \{c_i\}_{i=1}^K$ is considered. By applying the SAX procedure to the elements of X we obtain a collection of strings $\tilde{\mathcal{G}} = \{\tilde{g}_i\}_{i=1}^N$ of length $\frac{q}{w}$, with characters belonging to $\mathcal{A}$. After the symbolization procedure, following the scheme depicted in Sec. 3.1, we employ a ML classifier that exploits the distance defined by SAX. It has been shown (Ding et al., 2008) that a simple 1-NN classification guarantees state-of-the-art performance in terms of classification accuracy with time-series data; hence in this work, 1-NN is adopted in combination with SAX.

For any new observation to be classified, 1-NN (like all k-NN classifiers) needs to evaluate distances with all the available training data, requiring large space to store the entire training dataset; to make the approach feasible for the problem at hand, we use *templates*, i.e. $s \in \mathbb{N}^+$ observations for each class that are chosen as representative of all the observations available in the training dataset; the motivation of having more than one template per class, is that, input signals may drastically change with different scenarios (like wearable devices located in different body positions or activities performed in different environmental conditions) which could not be captured without using multiple templates.

The classification model is based on a set of $\mathcal{M} = \{m_{ij} \in \mathcal{A}^{1\times w}, i = 1, \dots, K, j = 1, \dots, s\}$. For each class, the s most representative templates can be found through a clustering technique based on SAX-distance, or, in a simpler fashion by selecting the most frequent s observations; in this work we adopt this last criterion.

In the case $s = 1$, the prediction task for a new observation x is easily performed by:

$$\hat{c}_i = \arg\min_i d(x, m_{i1}), \quad i = 1, \dots, K, \tag{4.4}$$

where $d := d_{SAX}$. When $s > 1$, given the premises on the choice for $s > 1$, a natural extension is given by:

$$\hat{c}_i = \arg\min_i d(x, m_{ij*}), \quad i = 1, \dots, K, \tag{4.5}$$

where, for the $\bar{i}$ class the optimal template is selected

$$j^* = \arg\min_j d(x, m_{\bar{i}j}), \quad j = 1, \dots, s. \tag{4.6}$$

Once gestures are classified, the focus must be placed on the AR (see Fig. 4.1). Activities are compositions of gestures: in order to move from gestures to activities, a sliding window approach is here employed. A fixed length window of width l_w is considered: in each window the predicted activity is chosen as the mode of the classified gestures; the window is then shifted forward of one gesture.

4.3 SAX Multivariate Extension

The framework proposed in the previous Section is limited by the fact that SAX technique only operates with 1-d time-series, which can lead to poor performance in AR problems. In this Section, we discuss extensions of the SAX approach to multidimensional cases.

Let assume of having N p-dimensional gestures $\mathbf{g}_i = [g_i^{(1)}; \ldots; g_i^{(p)}] \in \mathbb{R}^{p \times q}$ where $g_i^{(\cdot)} \in \mathbb{R}^{1 \times q}$. We define $d^{(i)}$ as the 1-dimensional SAX distance independently computed for the i-th dimension; we also define the multivariate distance $\boldsymbol{d} = \Psi(d^{(1)}, \ldots, d^{(p)})$. In the following, three different versions of $\mathbf{d}$ are presented:

$$\boldsymbol{d}_{\text{min}} = \min\{d^{(1)}, \ldots, d^{(p)}\} \tag{4.7}$$

$$\boldsymbol{d}_{\text{mean}} = \frac{1}{p} \sum_{i}^{p} d^{(i)} \tag{4.8}$$

$$\boldsymbol{d}_{\text{geom}} = \sqrt[p]{\prod_{i=1}^{p} d^{(i)}}. \tag{4.9}$$

With a trivial extension, we employ the just defined multivariate distances to classify accordingly to Figure 4.1[1].

The motivation behind distances (4.7) and (4.9) is that we want to discard, as much as possible, disturbances derived from changes in orientation; they act similarly with the difference that $\boldsymbol{d}_{\text{geom}}$ is less biased towards the minimum. These two measures could be unstable, especially where the intra-class and inter-class variance of distance are similar (i.e. when having low discrimination power). Distance (4.8) avoids this issue but could cause a convergence "phenomenon" where multivariate gestures tend to be equally distant.

4.4 Activity Identification and Gesture Extraction

In this Section we illustrate a procedure to extract gestures from streams of data and to automatically identify non-activities (AOM) regions.

The adopted approach is based on the following assumption: if a non-stationary input signal exhibits periodical (or quasi-periodical) behavior, it can be decomposed into intrinsic mode functions that represent the signal at different time scales. Thus, the *fundamental* mode functions can be extracted in order to capture the activity periodicity. The previous task can be accomplished by the well-known Empirical Mode

[1]Rigorously speaking, one should select multivariate templates by choosing the s most frequent vectors $\left\{\left(m_{1j}, \ldots, m_{pj}\right)\right\}_{j=1}^{s}$. However, this multivariate space is heavily sparse and this choice is affected by the typical "curse of dimensionality" issue Hastie, Tibshirani, and Friedman, 2009 that lead, in our simulations, to the choice of poor representatives for the classes. To overcome this issue, in this work we select templates independently for each dimension.

Decomposition (EMD) (Huang et al., 1998b) technique, however, for computational limits, this technique is not suited for the application at hand.

In this work, we employ a simpler filtering method that follows the aforementioned philosophy, a Gaussian filter with kernel $G_\sigma(t) = \exp[-t^2/(2\sigma^2)]/(\sqrt{2\pi}\sigma)$. Let $x(t)$ be the original and the $y(t)$ its filtered version, defined as

$$y(t) = \int_{-\infty}^{\infty} x(t-\tau)G_\sigma(\tau)d\tau \approx \int_{t-3\sigma}^{t+3\sigma} x(t-\tau)G_\sigma(\tau)d\tau, \tag{4.10}$$

where σ should be opportunely chosen with *a-priori* knowledge of the minimum fundamental frequency of $x(t)$. In fact, the relative cut-off frequency is $f_c = 1/2\pi\sigma$, while in the discrete domain (now σ is measured in samples), the cut-off frequency (in physical units) can be calculated from $f_c = F_s/(2\pi\sigma)$, where F_s is the sampling frequency.

Algorithm 3 Gesture extraction algorithm

```
1:  procedure GESTUREEXTRACTION($x(t)$, $\sigma$, $q$):
2:      Apply Gaussian filter $y(t) = x(t) * G_\sigma(t)$
3:      Event Identification phase. Return $y'(t)$, defined as $y(t)$ without AOM regions.
4:      Find local maxima points $\{P_i\}_{i=1}^N$ from $y'(t)$
5:      for all $P_i$ do:
6:          Find borders $(b_{1i}, b_{2i})$ , $b_{2i} > P_i > b_{1i}$
7:          Extract gesture $g_i$ delimited by borders $b_{1i}$ and $b_{2i}$
8:          Resample to $q$ samples
9:      end for
10:     return $\mathcal{G} = \{g_i \in \mathbb{R}^{p\times q}\}_{i=1}^N$
11: end procedure
```

Algorithm 4 Activity recognition algorithm

```
1:  procedure TRAINING($\mathcal{G}_{\text{train}}$,$\boldsymbol{\theta}$):
2:      Symbolize gestures in $\mathcal{G}$ and return $\tilde{\mathcal{G}}$
3:      Compute templates:
            $\mathcal{M} = \{\boldsymbol{m}_{ij}, i = 1,\dots,K, j = 1,\dots,s,\ \boldsymbol{m}_{ij} \in \mathcal{A}^{p\times w}\}$,
        where $\boldsymbol{m}_{ij}$ is the $j$-th most frequent gesture for class $i$
4:      return $\mathcal{M}$
5:  end procedure
6:  procedure CLASSIFICATION($\mathcal{G}_{\text{test}}$, $\mathcal{M}$, $l_w$):
7:      for all $g_i \in \mathcal{G}_{\text{test}}$ do
8:          Symbolize $g_i$ into $\tilde{g}_i$
9:          Predict gesture through 1-NN classifier:
                $\hat{c} = \arg\min i, \boldsymbol{m}_{ij} \in \mathcal{M} d_{SAX}(\tilde{g}_i, \mathcal{M})$
10:     end for
11:     Predict activities using a sliding window $l_w$
12: end procedure
```

After filtering, gestures are extracted as follows:

- first run a EI procedure in which non-AOM regions are detected; we accomplish this task by analyzing the auto-correlation of $y(t)$ on a sliding window over a reference univariate signal and by monitoring the overcoming of an opportune threshold η_{th} on the *fundamental* correlation peak. A dedicated set of experiments have been performed to choose a optimal value for η_{th}: as a result, in this work we adopt $\eta_{th} = 0.5$.

- a peak detection algorithm discovers maxima, which represent a one-to-one relation with each gesture;
- then, starting from each point of maximum, progress backwards and forward until two points of local minimum are found. These points are the borders of the examined gesture.

This procedure returns a collection of gestures of variable lengths. In order to make gestures comparable, i.e. invariant to *time warping*, we resample to a fixed length q: in this thesis we use the Akima interpolation (Akima, 1970). For the sake of clarity, Algorithm 3 and 4 summarize the fundamental AR steps.

4.5 Experiments

The proposed methodology has been tested on two different datasets:

- *HAR dataset* - a reduced version of the public UCI Human Activity Recognition (HAR) Using smart-phones Dataset (Anguita et al., 2013) where 3 continuous-repetitive normal day activities are considered: walking (WLK), walking upstairs (WUS) and walking downstairs (WDS). We are therefore examining a 3 classes AR problem. The dataset includes experiments that were carried out by 30 people where all the participants were wearing a smartphone (a Samsung Galaxy S II) on the waist during the experiment; 3-axial linear acceleration and 3-axial angular velocity have been captured at a constant rate of 50Hz using the embedded accelerometer and gyroscope of the device.
- Cross-Country Skiing dataset (*XC dataset*)- a private dataset of Cross-Country Skiing where 3 different styles were performed by 8 skiers; the three styles are

  1. double poling (DP), where both poles are used in parallel by the skier;
  2. diagonal stride (DS), where the poles are used in succession;
  3. kick-double-pole (KDP), a variant of DP, where an asymmetrical kick is performed by the skier.

  Athletes were wearing a smart-watch placed on the wrist; 3-axial linear acceleration, angular velocity and magnetic field, have been captured at a constant rate of 100Hz using the embedded accelerometer, gyroscope and magnetometer of the device.

Here we focus on 3D-acceleration signals ($p = 3$), which best capture signal variability. Furthermore, we set the fixed length of each gesture to $q = 150$ samples for both the datasets. For tuning purposes and in order to assess the algorithm performance, a cross-validation procedure on parameters $\boldsymbol{\theta} = (w, \alpha, s)$ is run. Moreover, the univariate classification (D1) of single dimension and the multivariate classification with the defined distances are compared: for the sake of simplicity, we will present only the results deriving from the dimension that guarantees the best accuracy (i.e. z-axis).

In the experiments, a L1-distance is also employed for accuracy comparison with the SAX-based classification: in this type of experiments, the classification is done directly on gestures, bypassing the symbolic representation based on a L1-distance. Templates, in this case, are chosen after a k-means clustering procedure Hastie, Tibshirani, and Friedman, 2009. Finally, for reasons of space, we shall report only the final AR results, which are our main concern.

Remark 1. Reported results was obtained following cross-validation Hastie, Tibshirani, and Friedman, 2009 paradigms in order to guarantee a fair evaluation of the proposed approaches.

In the following, for each dataset, results are reported following an accuracy improvement order.

Results of HAR dataset

In Fig. 4.2 we report the results for the univariate and multivariate AR problems for the SAX and L1 models, respectively. In particular, Fig. 4.2a, Fig. 4.2b, Fig. 4.2d and Fig. 4.2e refer to the HAR dataset while Fig. 4.2c and Fig. 4.2f refer to the XC dataset. The results for the 3-classes problem are unsatisfying and this is caused by the fact

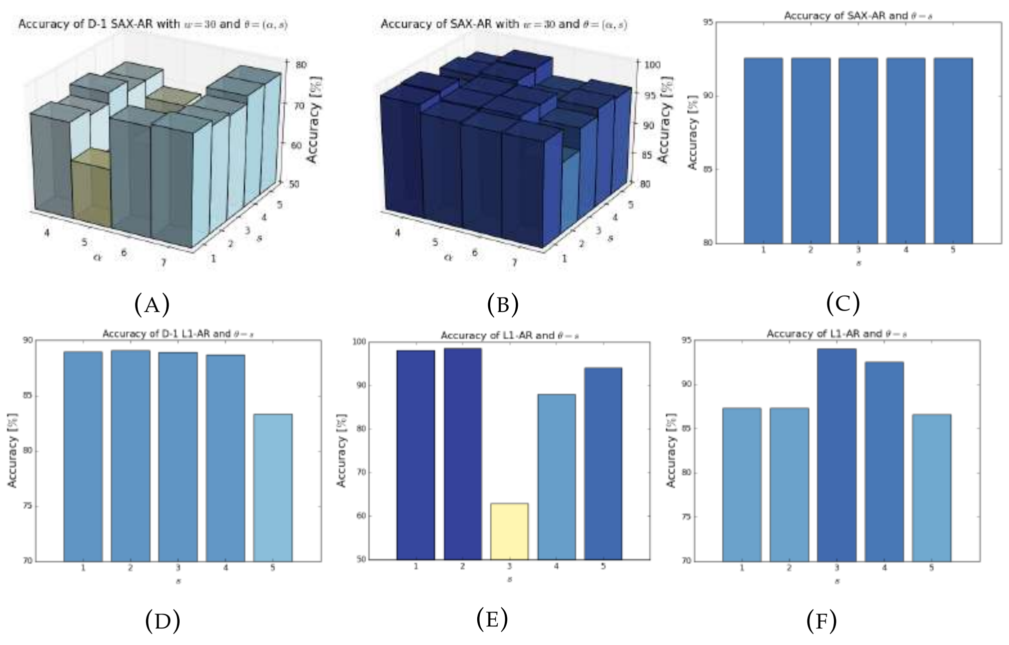


FIGURE 4.2: *Results of AR*. Comparison between univariate and multivariate approaches. Panels Fig. 4.2a, Fig. 4.2b, Fig. 4.2e, Fig. 4.2d refer to the *HAR dataset* while Panels Fig. 4.2c, Fig. 4.2f refer to the *XC dataset*.

that WDS and WUS are hardly distinguishable analyzing only z-axis, which is the axis towards the motion direction. Thus we consider a new binary problem with classes WLK vs WS (i.e. WDS and WUS are treated as a single class, WS).

As demonstrated, limiting the analysis to *one* axis could be too restrictive; in the following, it will be shown how the performance can be improved by employing the multi-variate extension presented in Sec. 4.3. All the distances (4.7)-(4.9) were tested and it was verified that the mean distance d_{mean} (4.8) performs systematically better then the others either in terms of robustness and of accuracy. Therefore, only the results for such distance are presented in the following.

Finally, in Table 4.1 we report the confusion matrices for SAX and L1 models at the optimal value of the tuning parameters θ, which represent the best cross-validation accuracy results for the univariate and multivariate problems, respectively.

Cross-Country Skiing dataset

As already stated, the original data at hand potentially allow the study of 3-classes recognition problem (DP, KDP, DS). However, classes DP and KDP can be considered

TABLE 4.1: *HAR dataset*. Confusion matrices (cross-validated results).

	Tab1a	Predicted (SAX)		Predicted (L1)	
		WLK	WS	WLK	WS
True	WLK	1293 (85.5%)	218 (14.43%)	1489 (98.54%)	22 (1.46%)
	WS	94 (12.53%)	656 (87.47%)	151 (20.13%)	599 (79.87%)

(A) Univariate problem with $\boldsymbol{\theta}_{SAX} = (\alpha, w, s) = (5, 15, 3)$ and $\theta_{L1} = s = 3$.

	Tab1b	Predicted (SAX)		Predicted (L1)	
		WLK	WS	WLK	WS
True	WLK	1504 (99.54%)	7 (0.46%)	1497 (99.07%)	14 (0.93%)
	WS	22 (2.93%)	728 (97.07%)	19 (2.53%)	731 (97.47%)

(B) Multivariate with $\boldsymbol{\theta}_{SAX} = (\alpha, w, s) = (6, 30, 2)$ and $\theta_{L1} = s = 2$.

indistinguishable given the data available: the gestures only differ by a kick that seems to be not 'observable' from the wrist, the location of the wearable collecting the data. Hence, we simplify the original classification problem in a 2-classes problem where DP and KDP are considered as the same class (K/DP). For the sake of conciseness, only the multivariate experiments are here reported (Fig. 4.2c and Fig. 4.2f), while cross-validated results are reported in Table 4.2.

TABLE 4.2: *XC dataset*. Confusion matrices (cross-validated results).

	Tab2a	Predicted (SAX)		Predicted (L1)	
		K/DP	DS	K/DP	DS
True	K/DP	81 (98.78%)	1 (1.22%)	81 (98.78%)	1 (1.22%)
	DS	9 (17.31%)	43 (82.69%)	7 (13.46%)	45 (86.54%)

(A) Univariate problem with $\boldsymbol{\theta}_{SAX} = (\alpha, w, s) = (7, 30, 1)$ and $\theta_{L1} = s = 3$.

	Tab2b	Predicted (SAX)		Predicted (L1)	
		K/DP	DS	K/DP	DS
True	K/DP	82 (100.0%)	0 (0.0%)	81 (98.78%)	1 (1.22%)
	DS	10 (19.23%)	42 (80.77%)	7 (13.46%)	45 (86.54%)

(B) Multivariate problem with $\boldsymbol{\theta}_{SAX} = (\alpha, w, s) = (7, 30, 2)$ and $\theta_{L1} = s = 3$.

4.5.1 Discussions

From the experiments reported in this work, it can be concluded that the proposed multivariate extension of SAX allows good classification accuracy for AR problems; this is achieved at a feasible cost for wearable devices in terms of complexity and memory. While the previous outcome is clearly proved in the case of the HAR dataset, for the XC dataset, results of multivariate and univariate problems are almost identical: in fact, all the informative content is contained in the x-axis, i.e. the axis which points to the direction of wrist motion (this was verified through a PCA procedures, indicating that the first Principal Component is represented by the x-axis). On the other hand, we experimentally verified that by increasing the number of templates s and, therefore, the complexity of the solution, accuracy does not systematically improves. We suppose that one of the reasons of this result is s should be set differently for each element of the set $R \times K$, where D is the dimension set and K is the templates set. For example, the optimal value $s = s_0$ for class k and dimension r, could be too low for certain classes/dimensions (which causes an under-estimation of complexity) or too high (which could cause the injection of noise).

Chapter 5

Unsupervised Model-Free Fault Detection in Artificial Pancreas

5.1 Articial Pancreas

5.1.1 Type I Diabetes and its Treatment

Type I Diabetes (T1D) is a disease caused by the autoimmune destruction of pancreatic β-cells. In healthy subjects, these cells are responsible for the secretion of insulin, a hormone playing a crucial role in the complex feedback mechanism that maintains blood glucose concentration (glycemia) tightly regulated despite large perturbations, such as meals and physical exercise (Cobelli et al., 2009). Subjects affected by T1D must therefore resort to external insulin administration and have to carefully choose insulin dosing to compensate for the impaired blood glucose (BG) regulation. Overdosing of insulin can induce hypoglycemia, BG<70 mg/dl, a condition posing an immediate threat for subjects' health and that can potentially result in seizure, coma and even death. On the contrary, persistent hyperglycemia, i.e. blood glucose above BG>180 mg/dl, is associated to long term complications, such as neuropathies, nephropathies and cardiovascular diseases (Control and Group, 1993; The Diabetes Control and Complications Trial and Epidemiology of Diabetes Interventions and Complications Research Group, 2000). Severe hyperglycemic episodes may also pose an immediate threat for subjects' health, especially if caused by hours-long absence of insulin (diabetic ketoacidosis).

Recent technological advances have come in help of T1D subjects in this extremely complex task: portable pumps for *Continuous Subcutanoeus Insulin Infusion (CSII pumps)* , shown on the right of Fig. 5.1, allow the subject to change insulin dosage throughout the day, also in real-time if needed (McAdams and Rizvi, 2016); minimally invasive subcutaneous glucose sensors, the so-called *Continuous Glucose Monitors (CGM)* , visible on the left of Fig. 5.1, collect frequent (1-5 min) BG concentration measurements without requiring finger pricks (McAdams and Rizvi, 2016); finally, in the last 10 years, closed-loop control systems modulating the insulin infusion on the basis of CGM reading, the so called *Artificial Pancreas (AP)* systems, have been developed (Thabit and Hovorka, 2016) and are becoming commercially available (*Medtronic Products Website: The MiniMed 670G Insulin Pump System.*).

Unluckily, both CGM sensors and CSII pumps are subject to failures, (Ramkissoon et al., 2017) that can be critical for the safety of the subject. This calls for the development of advanced monitoring systems, effectively detecting anomalies and malfunctioning by leveraging on the unique availability of rich data offered by the above technologies.

Of special interest for their potential impact, are faults affecting insulin pumps. Such faults can be due to mechanical defects (Guilhem et al., 2009) or to kinking,

FIGURE 5.1: A commercial subcutaneous glucose sensor (CGM, on the left) and a commercial insulin pump (on the right). The pump is connected to a subcutaneous needle via a catheter.

occlusion, and simple pulling out of the pump catheter from the insertion site (Schmid et al., 2010; Bon, Dragt, and DeVries, 2012). The reduced or missed delivery of insulin typically induces an hyperglycemic event and increases the risk of diabetic ketoacidosis (Guilhem et al., 2009). In this chapter we focus on this type of faults, that is not only the most dangerous for the patients but is also more challenging to be detected than other types of fault (Del Favero et al., 2014).

5.1.2 Literature Approaches to Detect Pump Faults in AP

Given the practical relevance of detecting faults affecting insulin pump, many works in literature dealt with this problem. Methods to detect pump faults based on heuristics and threshold values were proposed in (Howsmon et al., 2017) and (Cescon et al., 2016), and tested on real data.

A natural approach to tackle this problem is via model-based techniques. In fact, from a control systems perspective, a T1D patient can be thought as a dynamical system whose output to be controlled is the glucose concentration $g(t)$. The output is affected by a control input, the insulin injection $i(t)$, and by a process disturbance, the meal $m(t)$, that has to be rejected. These three signals, are available in current AP systems[1] and can be used for model-based fault detection schemes.

Facchinetti et al. (Facchinetti et al., 2013) proposed a fault detection method that, based on individualized linear black-box models of glucose-insulin interactions, builds a Kalman predictor of future glucose and checks if the measured glucose falls within the prediction confidence intervals. Del Favero et al. (Del Favero et al., 2014) extended such method domain from night-time to the whole day, considering also the effect of the meals.

Other approaches employed different nonlinear physiological models to perform predictions. A strategy based on the mechanistic physiological model developed by Hovorka et al. (Hovorka et al., 2004) was formulated by Vega-Hernandez et al. (Vega-Hernandez et al., 2009), and tested in two simulated scenarios of actuator faults. A fault detection scheme based on modal interval analysis was applied by Herrero et al. (Herrero et al., 2012) to a modified version of the physiological model developed by Bergman et al. (Bergman, 1989) and was tested on simulated data to detect pump faults. Kovacs et al. (Kovács et al., 2006) transformed the Bergman et al. model in affine parameter-varying form in order to handle the dynamics and to estimate unknown disturbance inputs of the system. Mahmoudi et al. (Mahmoudi et al., 2016)

[1]An estimate of $m(t)$ is provided by the patient at each meal and used to compute a feed-forward control to better compensate postprandial glucose increase.

provided an analysis of the performance of three nonlinear filters (Extend Kalman Filter, Unscented Kalman Filter, Particle Filter), operating on the Medtronic Virtual Patient model (Kanderian et al., 2009), for fault detection in CGM.

The performance of the above approaches, and more generally of any model-based technique, are strongly dependent on the quality of the model of which they leverage. Unluckily, having access to accurate models of T1D physiology is non trivial. In fact, the physiology of each individual is significantly different from that of other subjects (inter-patient variability), and even the same patient physiology changes over time (intra-patient variability). This makes the usage of fixed, average or population models without customization on the subject and periodic updates, hardly viable. Unfortunately, identifying an accurate model for a specific patient having access only to $g(t)$, $i(t)$ and $m(t)$ is still a challenge. Unfortunately, identifying an accurate model for a specific patient having access only to $g(t)$, $i(t)$ and $m(t)$ is still a challenge, (Oviedo et al., 2017).

An alternative, model-free, strategy is to employ data-driven classification approaches exploiting the availability of historical data. These methods can be divided in *supervised* or *unsupervised* depending on whether they require or not the availability of labeled data, i.e. data where the information 'Fault'/'Non Fault' is present. To the best of our knowledge, the only contribution that explored data-driven approaches is (Rojas, Garcia-Gabin, and Bequette, 2011), where Rojas et al. used *supervised* methods by training a classifier to detect insulin pump faults.

A drawback of supervised approaches is that labeled datasets are hardly available. In fact, in month-long data collection during which a patient is using the AP, the true functioning/faulty status of the pump is unknown. To reconstruct it, expert human operators have to manually label the dataset with time-consuming a posteriori visual inspections, in a challenging procedure prone to errors. Also, this procedure can only be done on a small subset of subjects, therefore subject-specific labeled data are in general not available for all patients.

This makes the use of unsupervised approaches particularly appealing for advanced monitoring problems.

5.1.3 Anomaly Detection: Model-Based vs. Data-Driven

In the context of Smart Monitoring, Anomaly Detection (AD) tools are fundamental instruments to detect anomalous behavior in data (Susto, Beghi, and McLoone, 2017). In many fields, the primary application of AD methods is to detect faults (Bastani et al., 2013) or frauds (Hamlet et al., 2017), and, in complex/multi-dimensional systems, AD-based tools are necessary to detect abnormal behaviour that cannot be captured even by domain experts (Susto, Beghi, and McLoone, 2017).

Anomalies are data patterns that have different data characteristics from normal instances (Beghi et al., 2014); a statistical terminology for anomaly is *outlier*, an observation/instance that is "different" from others, while, on the other hand, non-anomaly observations are referred as *inliers*. As hinted before, AD methods can be categorized into two families: model-based and data-driven approaches. In this work, we deal with the latter category given the advantages detailed before. We underline that models of multi-dimensional and/or dynamical systems could be costly, challenging, or even impossible to be accurately defined and, partially for this reason, model-free approaches are generally a desired feature in many AD applications (Broderick, Allen, and Tilbury, 2011). This is especially true in the application considered here.

In the field of data-driven algorithms, a classical monitoring approach is the employment of univariate control charts (Jensen et al., 2006), where the crossing of predefined control limits trigger alerts/warnings. However, one of the main criticisms of monitoring a system only through control charts, is that multivariate aspects are not captured. Moreover, mixture distributions are difficult to be automatically monitored with such tools.

Nowadays, with the increasing availability of data in every aspect of our society, the need for multi-variate, high-dimensional *Anomaly Detection/Isolation* techniques is thriving; instruments to implement efficient ADs are provided by machine learning (ML). ML approaches have proliferated in recent years for ADs in many sectors like building automation (Cheng et al., 2016), manufacturing (Susto, Terzi, and Beghi, 2017), security (Hamlet et al., 2017) , and wearable (Veeravalli, Deepu, and Ngo, 2017); this was achieved thanks to the algorithmic advancements in the field and the increased computational and storage capabilities of IT infrastructures and portable devices (Barazandeh et al., 2017).

In this work, we employ modern ML approaches, *Local Outlier Factor (LOF), Connectivity-based Outlier Factor (COF), Isolation Forest (iF/iForest)*, to detect outliers; moreover, we adapt such approaches, generally thought for 'static' problem, to problems that have a time-series evolution: the proposed framework is here called *4TSD (For Time Series Data)*.

5.1.4 Contributions and Organization

In this chapter, we tackle the problem of detecting insulin pump faults in an Artificial Pancreas with a novel approach that relies on advanced data-driven anomaly detection algorithms. The main contributions are:

- Applying a model-free and unsupervised approach to monitor the AP. The use of a model-free approach allows to avoid the complex sub-step of identifying a model. The usage of an unsupervised approach in contrast with supervised ones, prevents the need of labelled data (hardly available in practice).
- Proposing the 4TSD adaptation of the AD methods. This modifies the advanced unsupervised model-free anomaly detection algorithm available in literature, designed to work on static feature sets, to better deal with the dynamical nature of the T1D physiology and other dynamical systems.
- Conducting an extensive validation of the proposed methods on simulated data, leveraging one of the most accurate tool currently available for computer simulation of T1D subjects, the "UVA/Padova Type 1 Diabetic Simulator" described in Sec. 5.3.
- Conducting a proof-of-concept validation on real-patient data collected during a prolonged, free-living experiment on the AP.

The reminder of this chapter is organized as follows: Sec. 5.2 reviews the AD methodologies adopted and introduces the 4TSD adaptation for time series data of such algorithms; moreover, in Sec. 5.2, the domain-expert designed features considered in this work are also detailed. Sec. 5.3 describes the datasets: SubSec. 5.3.1 illustrates the simulation environment ("UVA/Padova Type 1 Diabetic Simulator") and reports the details of the virtual experiment performed, while Sec. 5.3.2 presents the real data and the processing requested to use them. The experimental results are

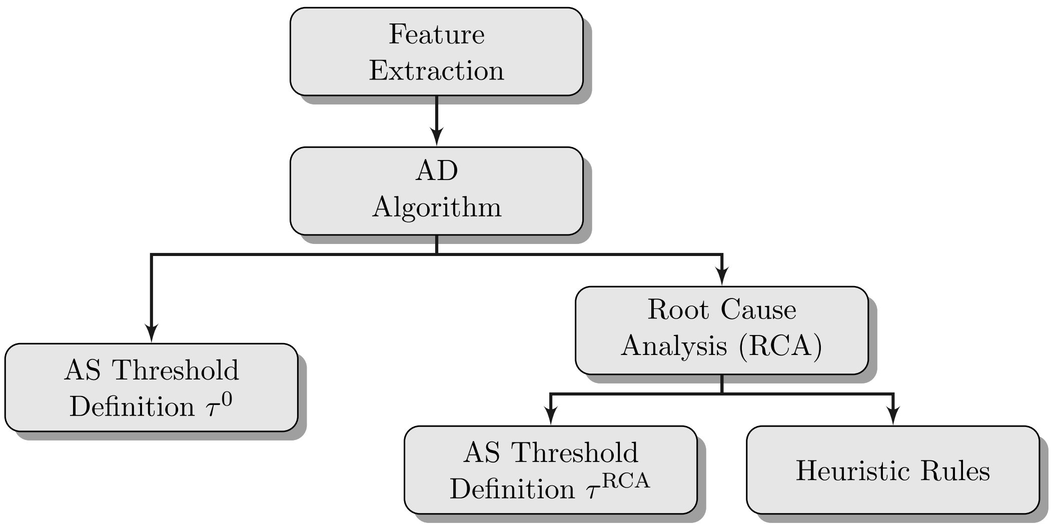

Feature Extraction
AD Algorithm
AS Threshold Definition τ^0
Root Cause Analysis (RCA)
AS Threshold Definition τ^{RCA}
Heuristic Rules

FIGURE 5.2: Main phases of the adopted anomaly detection procedure.

presented and discussed in Sec. 5.4 (virtual data) and Sec. 5.5 (real data). Sec. 5.6 discusses some limitations of the proposed methods and outlines future developments.

5.2 Methods and Features

5.2.1 Anomaly Detection Procedure

The standard phases of an AD procedure are:

1. *Feature Extraction* – This stage involves the definition of *features*, quantities that describes in a concise way the informative content of the underlying signals. The choice of a proper feature set is fundamental in order to build a rich feature space in which the informative content of the raw signals is captured and possibly, anomalies are distinguishable from normal instances. In the definition of the feature space it is generally fundamental to involve domain experts;

2. *AD algorithm* – Once the features are defined and extracted, AD algorithms are applied and compared. Each AD algorithm returns an Anomaly Score (AS) for each data point: an AS is a continuous value that characterizes the 'degree of anomaly' of a data point;

3. *Anomaly Score (AS) Threshold definition* – In an automatic monitoring system, actions/alarms need to be associated with certain values of the AS. Here, a typical approach of defining threshold on the AS is adopted: once the AS oversteps such predefined threshold, an anomaly is flagged. The criterion for the choice of the threshold τ^0 is dependent on the particular performance metric adopted, that is generally a trade-off between False Negatives (FN) - wrongly identified as inliers - and False Positives (FP) - data points wrongly identified as outliers. More details will be provided in Sec. 5.3.

The conceptual elements of the AD procedure adopted in this work is illustrated in Fig. 5.2. It can be appreciated how a Root Cause Analysis (RCA) module is present in the scheme; before going into the details of such procedure, a remark regarding the employed fault detection solution is due. The most powerful characteristic of this approach is that it is *unsupervised*; as mentioned earlier, this leads to several advantages, the most important being that no labeling procedures are required to build the model. However, this comes at cost of sacrificing discriminative power of supervised models: this means that our models find all the anomalous phenomena regardless whether they are the fault under exam or not. In this work we report worst case scenario results: all the identified outliers are reported as FP if no pump faults are present, even if, from a statistical point of view such data point is in fact an anomaly.

The aforementioned issue can be mitigated using prior information and a RCA procedure. During the Root Cause Analysis (RCA), a deeper look at FP instances has been made and we noticed that many FPs were in correspondence of relatively rare situations such as hypoglycemia, that were not due to pump faults. We therefore applied a procedure that "silence" the AS in time regions where glycemia is under a given value. In turn, other than reducing FPs, this allows us to lower the anomaly threshold and, then, possibly detect more true positives (TPs). As represented in the right branch of the diagram of Fig. 5.2, the aforementioned procedure was adopted in this work, leading to the heuristic rule for dealing with potential hypoglycemia situations and the definition of a new threshold τ^{RCA}; both AD approaches will be compared in the following.

5.2.2 Anomaly Detection Algorithms

Before illustrating the methods used in this work, we define the notation: $x \in \mathbb{R}^p$ is an observation of x (an individual data point) a set of p features characterizing the problem. Let N indicate the number of points populating the dataset, denoted with $X = \{x(t),\ t = 1, \dots N\}$. Finally, let k indicate the cardinality of the neighborhood $\mathcal{N}_k(x)$.

The unsupervised methodologies we investigate in this chapter are: (i) *Local Outlier Factor* (LOF) (Breunig et al., 2000), (ii) *Connectivity-based Outlier Factor* (COF) (Tang et al., 2001) and (iii) *Isolation Forest* (iF/iForest) (Liu, Ting, and Zhou, 2008). LOF and COF belong to the family of *density*-based methods. Such approaches are based on the study of local neighborhoods: an observation in a dense region is considered an inlier while a data point in a low-density region is considered as an outlier. The three methods are now illustrated in the following.

[i] The basic idea of LOF is comparing the density of a point w.r.t. its neighbors instead of considering all the points in the dataset; by doing so it is possible to identify regions of similar densities and point (flagged as outliers) with significant lower density than its neighbors.

LOF involves two steps:

1. evaluating the so-called Local Reachability Density (LRD);
2. computing the anomaly score s_{LOF}.

The LRD of a data point x in the $k-$Neighborhood $\mathcal{N}_k(x)$ is defined as:

$$\text{LRD}_k(x) = \frac{k}{\sum_{z \in \mathcal{N}_k(x)} r_k(x,z)}, \tag{5.1}$$

where $r_k(x,z) = \max\{d_k(x), d(x,z)\}$ is the reachability distance and $d_k(x)$ is the distance from x of its k-nearest neighbor. The reachability distance $r_k(x,z)$ is used instead of $d(x,z)$ in order to reduce statistical fluctuations in the computation of the s_{LOF}.

It is straightforward to see that, when the set of coefficients $\{r_k(x,z)\}_{z\in\mathcal{N}_k(x)}$ are (on average) low, $\text{LRD}_k(x)$ is high while, when neighbor points are far, $\text{LRD}_k(x)$ is low.

The correspondent anomaly score s_{LOF} is defined as:

$$s_{\text{LOF}}(x;k) = \frac{1}{k} \sum_{z\in\mathcal{N}_k(x)} \frac{\text{LRD}_k(z)}{\text{LRD}_k(x)} \in [0,\infty]\,. \tag{5.2}$$

The previous has to be interpreted as a *local* anomaly: a value of approximately 1 indicates that the data point is comparable to its neighbors (and thus not an outlier). A value below 1 indicates instead a denser region (which would be an inlier), while values significantly larger than 1 indicate outliers.

LOF is computationally expensive as it requires to find the nearest k neighbors for each observation, and for the observation in its neighborhood: thus the complexity[2] is $O(N^2)$. For low-dimensional data, the complexity can be reduced to $N\log N$ (on average) using a k-d tree algorithm (Bentley, 1975).

[ii] LOF, and density-based algorithms in general, perform well when data are distributed (as near) as isotropic clusters, but that they are not able to capture low-dimensional and complex data structures . In order to overcome this issue, Connectivity-based Outlier Factor (COF) was introduced (Tang et al., 2002). The main contribution with respect to LOF is that it tries to capture if connected regions (in the neighborhood of x) exist and computes the anomaly scores with respect to them. More concretely, the difference between LOF and COF stands on the ability of the latter to deal with cases where also low-density regions present a structure which is assumed to be "not-anomalous". For example, in a 2-D feature space, a line is an example of low-dimensional structure. Thus, assuming that laying on the line is not an anomalous behavior while points "outside" the line are outliers. In this case, LOF would fail since it is a purely density-based approach while COF allows to implicitly "learn" the normal structure using connectivity-based reasoning. The complexity is $O(N\log N)$ on average (with medium size datasets). We refer the interested reader to the original paper (Tang et al., 2002) for more details on COF.

[iii] The last AD method considered in this work is iForest[3], that is based on the simple assumption that anomalous points x_i must be few and isolated. The main idea of this method is based on the concept of space partitioning: an isolated point (anomaly) requires (on average) less iterations than an inlier to be isolated through partitioning.

This partitioning procedure (for isolation) can be implemented by a particular binary tree, called *iTree*, that is the result of a random partitioning procedure obtained by splitting the data based one of their feature at each iteration of the algorithm. Thus, we expect that, in the case of anomalous points, the path to reach a leaf node is shorter than the one for a normal point. However, a single tree can give an estimate of the path length which has high variance. Thus, similarly to Random Forest, an ensemble of T trees is constructed in order to provide low-variance estimation. More in detail, an iTree is built as follows:

1. a sub-sample of data $S \subset X$ is randomly selected;

2. a feature $v \in \{1,\ldots,p\}$ is randomly selected and a node in the iTree is created where the value of v is evaluated;

[2]Here we drop the dependency of p as it is considered small and fixed.

[3]Isolation Forest is inspired and should not be confused with the popular supervised method Random Forests.

3. a random threshold x_v^* is chosen in the domain $\{x_v^{min}, x_v^{max}\}$ of variable v;
4. two children nodes are created: the left node corresponds to points with $x_v < x_v^*$, the right one to $x_v \geqslant x_v^*$;
5. procedures from 2 to 4 are repeated until, either a data point is isolated, or the height limit $\log_2 |S|$ is reached[4].

The training stage requires $O(T|S|^2)$.

Then, the anomaly score $s_{\mathrm{IF}}(x)$ is computed for every point $x \in X$ (evaluation phase). This is done by evaluating the length $h(x)$ of the path from the root to x for each tree and computing

$$s_{\mathrm{IF}}(x) = 2^{-\frac{Eh(x)}{c}} \in [0,1]\,,$$

where $Eh(x)$ is the average path-length $h(x)$ on the forest and c is an adjustment factor which is set to the average path length of unsuccessful searches in a binary search tree procedure. When $Eh(x) \gg c$, the anomaly score approaches 0, meaning that x is considered a normal instance. On the other hand, when $Eh(x)$ is small, the $s_{\mathrm{IF}}(x)$ approaches 1, meaning that x appears to be an anomaly. Other than the advantages already exposed, iForest does not utilize distances or density measures to detect anomalies: this eliminates a major computational cost of distance calculation. Moreover, for each data point, the evaluation stage has linear time complexity $O(T|S|)$.

The parameters to be tuned in iForest are:

- the sub-sample size ψ of S, which can remain relatively small;
- the number of trees T, which should be enough to allow path length convergence.

The original paper of iF (Liu, Ting, and Zhou, 2008) reports as typical values for ψ and T, 100 and 256 respectively.

Furthermore we considered other anomaly detection methods such as Angle Based Outlier Detection (ABOD) (Kriegel, Zimek, et al., 2008), and Over-Sampling PCA (OSPCA) (Yeh, Lee, and Lee, 2009). In our preliminary analysis they were clearly outperformed by the methods presented here and hence we exclude them from our comparison.

5.2.3 Anomaly Detection for Time-series Data

The methods presented above, are usually derived in a set-up where the features are independent from each other. On the contrary, our problem involves time-series describing the evolution of biological systems in which features at a given time depend strongly on the previous ones. This means that features "close" in time are very likely close also in the feature space. As a consequence anomalous points will not be completely separated from normal points, but they will be instead connected to the cluster of normal points by thin, tail-shaped, structures, see Fig. 5.3.

Both isolation-based and density-based methods are challenged in the anomaly evaluation by the presence of these tails, that make the anomaly less isolated in the feature space. This behaviour is identified in AD literature as the *masking effect* (Liu, Ting, and Zhou, 2008).

[4]Such value for the height limit is chosen as it is the average length of a binary tree.

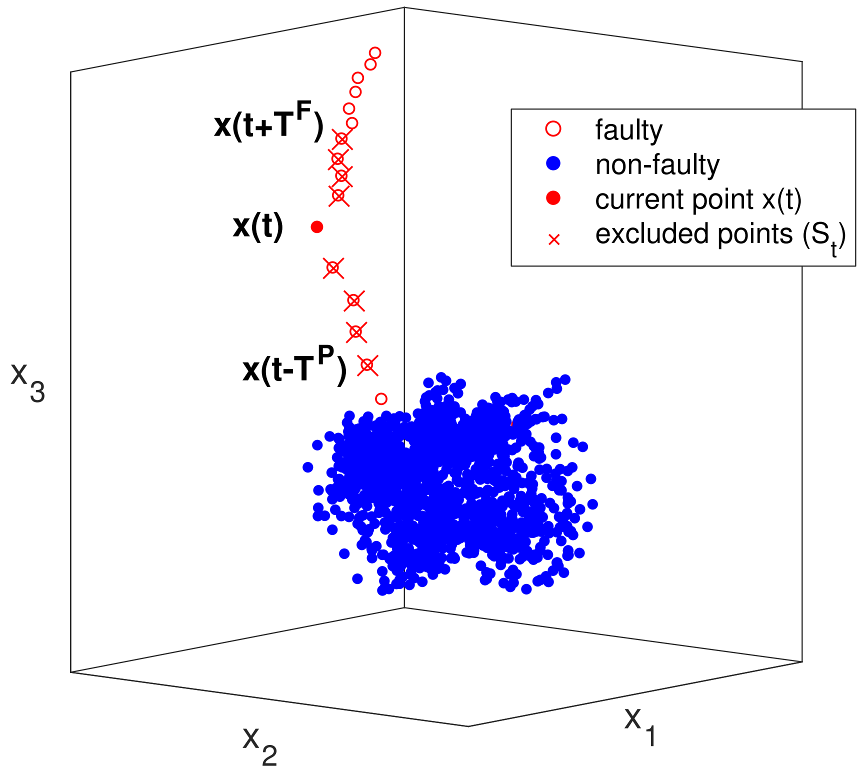


FIGURE 5.3: Illustration of the functioning of 4TSD adaptation

To ease this problem, when evaluating an observation $x(t) \in \mathbb{R}^p$, we propose to remove the previous T^P points and the future T^F points from the dataset, i.e. evaluate x(t) using $X \backslash \mathcal{S}_t$ instead of X, with

$$\mathcal{S}_t = \{x(t - T^P), \dots, x(t-1), x(t+1) \dots, x(t + T^F)\}.$$

In such a way, normal data points will be still embedded in a clustered region, while the anomalous points will be more isolated and possibly far from the cluster, see Fig. 5.3. The parameters T^P and T^F, are tuned in the experimental phase.

We applied this modification, called "procedure For Time Series Data" (4TSD), to LOF and COF, see Algorithm 5.

Algorithm 5 Algorithmic representation of 4TSD adaptation for LOF. Identical for COF.

```
Require: $X \in \mathcal{R}^{N \times p}, T^P, T^F$
1: for $t = 1, \dots, N$ do
2:     Select the current point $x(t) \in X$
3:     Compute $s_{\text{LOF}}(x(t); k)$ on restricted dataset $X \backslash \mathcal{S}_t$
   $\mathcal{S}_t = \{x(t - T^P), \dots, x(t-1), x(t+1) \dots, x(t + T^F)\}$
4: end for
5: Return $\mathbf{s}_{LOF} = \{s_{\text{LOF}}(x(1); k), \dots, s_{\text{LOF}}(x(N); k)\}$
```

Regarding iForest, the application of the 4TSD adaptation is not expected to bring any improvement, since iF is already designed to handle masking effects by building the isolation trees on a sub-sampled dataset S of size ψ, (Liu, Ting, and Zhou, 2008). In our case both ψ and the number of excluded points $|\mathcal{S}_t|$ are much smaller than N.

As a result, sampling ψ points from $X \backslash \mathcal{S}_t$ rather than X has little impact. For this reason, we don't present results of 4TSD for iForest.

5.2.4 Online Evaluation and Training

It is apparent that AD approaches have to be exploited on-line in the application at hand. The computational burden of the proposed approaches is compatible with on-line usage, i.e. it is possible to perform the evaluation of the AS when a new data-point becomes available in an acceptable amount of time.

As a matter of fact, thanks to the relatively large sampling time in AP systems, one viable possibility is to consider a 'brute force' implementation of the proposed AD methodologies, that consists in updating all anomaly scores on all data whenever a new data sample becomes available. In Sec. 5.4.1 we will evaluate the computational time of this 'brute force' implementation on a 30 days dataset, a representative size for the problem at hand. As illustrated in Table 5.2, the computational time is in the worst case smaller than 40s, and hence compatible with real-time use, since new data are typically available every 5 minutes in AP systems.

Moreover, less computational intensive implementations can be adopted. Let us first consider iForest. In this case, the AS score of a new data point can be evaluated without recomputing the whole AD model. The time for evaluating the anomaly score is completely acceptable since its computational cost is $O(\log \psi)$ for each tree. Regarding LOF and COF, an additional complexity has to be considered, since simple evaluation cannot be performed without re-training of the model. However the time burden reported in Table 5.2 can be abated by employing online versions of COF and LOF. Such versions provide solutions constructed in an incremental fashion each time a new data is available (Pokrajac, Lazarevic, and Latecki, 2007; Salehi et al., 2016).

5.2.5 Feature Extraction

As depicted in Fig. 5.2, a Feature Extraction procedure is necessary to apply the AD methods considered in this work. Feature Extraction can be done exploiting domain expert knowledge or be performed in a data-driven manner (Susto et al., 2016). Here, we opted for the domain expert approach, and the rationale behind the two-step design procedure adopted is described in the following.

Features Accounting for Dynamics

As previously mentioned, a T1D patient can be thought as a dynamical system whose output to be controlled is the glucose concentration $g(t)$. The output is affected by the control input, insulin injection $i(t)$, and the disturbance to be rejected, the meal $m(t)$, illustrated in Fig. 5.4a. Therefore a potential set of features describing the status of system at time t could be the quadruple $[g(t), \dot{g}(t),\ i(t),\ m(t)]$.

However, insulin injected at a certain time t has little or no impact on the behavior of the system at the same t, but rather on the behavior of the system in the next [1-8] hours. This is due to the slow absorption dynamics and to the persistence of past insulin, whose concentration exponentially decreases in the next 4-8 hours. To account for such dynamics, diabetologists introduced the concept of Insulin-On-Board (IOB), that is a rough estimate of residual insulin present in the blood at a given time. Mathematically, IOB is obtained by convolution of the injected signal $i(t)$ with an exponentially decaying function, as defined in (Ellingsen et al., 2009). Similarly, meals ingested at a certain time have no impact on the behavior of the system at the same time, since meals are absorbed in 15-45 minutes and then gradually fade in $\approx 4/6$ h. Hence, we introduce the Carbohydrates-On-Board (COB), also defined as the convolution of $m(t)$ with a suitable exponentially decaying function provided in (Schiavon et al., 2014). IOB and COB are illustrated in Fig. 5.4b. The first set of features describing our system and accounting for the physiological dynamics is therefore:

$$x_d(t) = [g(t), \dot{g}(t),\ \mathrm{IOB}(t),\ \mathrm{COB}(t)].$$

Optimized Features for Pump-Faults Detection

Since the aim of this chapter is to detect insulin pump malfunctioning, instead of considering the generic 4-dimensional feature set $x_d(t)$, describing the state of the system at time t, we introduced a nonlinear transformation of the feature set aimed to better highlight the anomaly of interest.

In particular, during a pump fault, blood glucose increases steadily despite the absence of COB. Therefore we introduce the feature

$$\mathrm{DCOB}(t) = \frac{\alpha \dot{g}(t)}{\beta \mathrm{COB}(t) + 1},$$

with α and β being two positive constants. $\mathrm{DCOB}(t)$ could be seen as a weighted glucose derivative: the weight decreases the importance of the glucose variation $\dot{g}$ when COB is large. In fact, in non-anomalous conditions, large COB values leads to large glucose variations. On the contrary, when $\mathrm{COB}(t) = 0$, $\mathrm{DCOB}(t)$ is simply proportional to $\dot{g}(t)$. Hence, large positive values of $\mathrm{DCOB}(t)$ occur when glucose increases in absence of carbohydrates on board and are symptoms of pump faults. α and β are used to magnify the impact of one signal over the other.

Furthermore, to keep patients in good metabolic control, new carbohydrates entering in the system should be compensated by extra insulin administration. The ratio between these two quantities changes as a consequence of control actions, for instance when extra insulin is delivered to decrease glucose. When a fault affects the closed-loop, the controller attempts to counteract glucose rise (without success, since insulin is not actually administered) and the alleged IOB increases significantly with respect to COB. In view of this, we introduce the feature

$$\mathrm{ICOB}(t) = \frac{\gamma \mathrm{IOB}}{\delta \mathrm{COB} + 1}, \tag{5.3}$$

with γ and δ being two positive constants. ICOB is a weighted version of IOB: the denominator in (5.3) decreases the importance of IOB when COB is large; in fact, simultaneous large values of this quantity are expected in well-controlled patients. On the contrary, the denominator in (5.3) does not penalize IOB when $\mathrm{COB}(t) = 0$, hence large positive values of ICOB are symptomatic of a large effort to lower glucose. Large unsuccessful attempts to lower glucose are symptoms of a pump fault. γ and δ are tuned to magnify the impact of one signal over the other.

Together with DCOB and ICOB , ad hoc designed for failure detection in pumps, we retained in the feature set the glucose concentration, $g(t)$, since large and prolonged values of this quantity are nowadays considered by T1D subjects as the main indicator for a potential pump occlusion.

In this work, we therefore adopt the features set

$$x(t) = [g(t), \text{DCOB}(t), \text{ICOB}(t)].$$

Constants value were tuned manually, we set $\alpha = \gamma = \beta = \delta = 10$. DCOB and ICOB are illustrated in Fig. 5.4c. Furthermore, a 3D plot of $x(t)$ collected in a 30 days simulation of subject #001 is depicted in Fig. 5.5a. In this example, faulty points, depicted in red, are visibly separated by non-faulty ones, depicted in blue.

5.3 Dataset

The algorithms proposed in this manuscript have been tuned and tested on two large synthetic datasets (N=100 virtual patients, 1-month), generated using a well-accepted computer simulator of T1D physiology, the so called UVA/Padova T1D simulator, (Man et al., 2014). These data are far from being a toy-example and the simulator was approved by the US Food and Drug Administration as a substitute of animal testing prior to AP clinical trials on humans. This tool has been broadly adopted in diabetes technology research. Furthermore, a proof-of-concept validation has been conducted on a real dataset (N=7 real patients, 1 month), collected during a real-life test of the AP, (Renard et al., 2016).

It should be noted that the data chosen to tune or test AD algorithms must contain accurate labels on the functioning/faulty status of pump, to be used as ground-truth to assess the methods. Unless dedicated experiments are performed (e.g. (Cescon et al., 2016; Howsmon et al., 2017)), real-data collected using AP system lack of this crucial information. The only option is retrospective label assignment via visual inspection. Even if performed by domain experts, this procedure lacks of both sensitivity and specificity. On the contrary, synthetic data allow to create faults in perfectly known positions and hence provide a perfectly accurate ground truth. Notably, such dataset can be generated without posing the safety of the patient at danger.

5.3.1 In-silico data

The UVA/Padova T1D simulator captures inter-patient variability (the large differences in the physiology of different T1D subjects) by providing a population of $N = 100$ adult subjects, whose physiological parameters are distributed to replicate the variability observed in T1D subjects. In this work, we use an updated version of the simulator (Toffanin et al., 2017), capable to account for variation, intra-patient variability (i.e. changes over time in a patient's physiology, (Visentin et al., 2015)).

In-silico protocol: for each subject, we simulated twice a 30 day protocol, with 3 meals per day, taking place with uniform probability in the intervals [06:30, 08:30] (breakfast), [12:00, 14:00] (lunch) and [19:30, 21:30] (dinner) with carbohydrates consumption uniformly distributed in [20, 60]g, [50, 90]g and [50, 90]g respectively. Throughout the day, insulin is administered to the patient by a PID closed-loop algorithm (basal insulin administration) with aim to keep patient blood-glucose in the nearly-normal range. At each meal, the patient delivers himself an additional insulin dose (insulin bolus), proportional to the estimated carbohydrates amount. We described the possible carbohydrates estimation errors performed by the patient as a Gaussian error with zero-mean and standard deviation 20% of the true meal amount. Blood glucose measurements and insulin data were available each $T_s = 5$ min, a typical sampling time in AP systems, for instance (Kropff et al., 2015).

The measurement error of CGM sensor is modeled as proposed in Facchinetti et al. (Facchinetti et al., 2014).

In the 30 days period we simulated $N_f = 1$ episode per subject of complete insulin pump fault, i.e. a complete undesired suspension of insulin delivery of which the control algorithm is not aware of, that creates a discrepancy between the commanded and injected insulin values. This frequency of the faulty event matches the incidence reported in literature, (Bon et al., 2011). In this work, we consider only suspensions occurring at night, as they are more challenging to detect than those blocking the large insulin boluses occurring during the day. The fault lasts for 6 hours after which we assume that the fault is detected and repaired by the patient.

For illustration purposes, a 3 day portion of data from patient #001 is reported in Fig. 5.4a. It can be noticed that, as a consequence of the malfunctioning, the controller is unable to counteract the raise of glycemia.

5.3.2 Real data

We considered a dataset collected during a clinical trial aimed to test the sustained use of AP in real-life (Renard et al., 2016). We will focus on the N=7 adult patients recruited in Padova, Italy. The patient uses the AP system for 24h a day over approximately 1 month. No protocol restriction was imposed to the patients, who were free to have meals and exercise as desired.

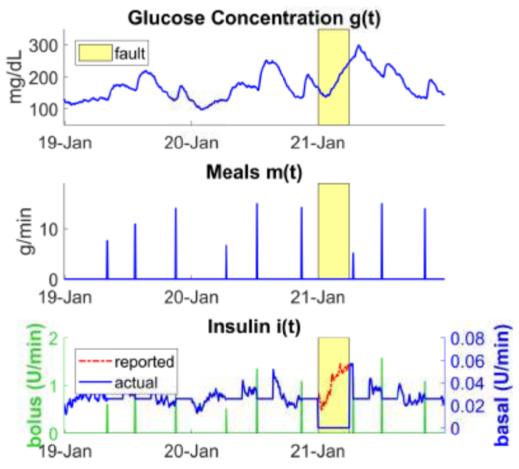
Unluckily, whether or not the pump was correctly functioning is unknown in this dataset. A further limitation of the dataset available is due to the fact that the patients were encouraged to respect the manufacturer guidelines for pump catheter replace (replacement every 3 days). Noncompliance with this 3-day replacement schedule is common among the patients and known to increases the likelihood of occlusions (Bon et al., 2011).

To address the first limitation, a team of two experts (an engineer and an clinician) who have been in charge of the trial (Renard et al., 2016), were asked to visually inspect the collected data and retrospectively label them. Due to the complexity of the task, a faulty/non-faulty labeling was not possible and was instead replaced with a four-class labeling, reported in Table 5.1.

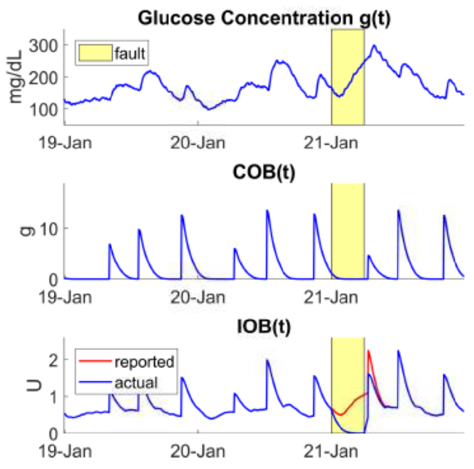
TABLE 5.1: Classes for retrospective data labeling

Label	Description
Fault	Anomalous hyperglycemia possibly due to a pump fault or caused by another issues producing similar effect (e.g. unreported CHO ingestion)
Other	Anomaly not due to a pump malfunctioning but rather to other issues, e.g. large CGM variation due to calibration
Unclear	Suspicious data portion lacking clear evidences to be classified as certainly anomalous
Not Fault	Non anomalous data

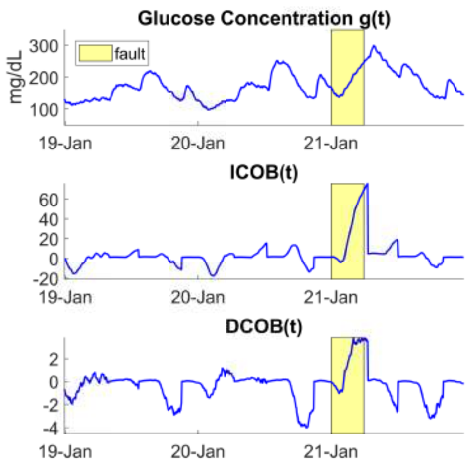
Beside this first dataset, obtained by retrospective labeling, a second dataset was generated. It was obtained in in two steps: (i) portions labeled as faulty were discarded and (ii) one fault per patient was introduced a-posteriori. This was done with the intention of removing uncertainties on the retrospective manual labeling.



(A) Original dataset: time evolution of $g(t)$, $i(t)$ and $m(t)$

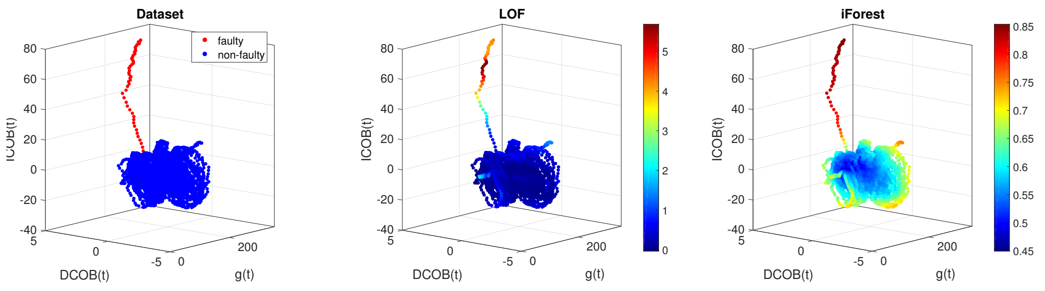


(B) Features accounting for dynamics: time evolution of $g(t)$, $IOB(t)$ and $COB(t)$



(C) Optimized features for fault detection: time evolution of $g(t)$, $DCOB(t)$, $ICOB(t)$.

FIGURE 5.4: A 3-day portion of data from the simulated patient #001. An insulin pump fault occurs around day 3 (highlighted by the yellow box). Left panel (a) displays unprocessed data, the central panel (b) displays the feature accounting for dynamics and the right panel (c) the feature designed to highlight the fault. The undesired drop in insulin levels causes a raise of glycemia. The controller attempts a counteraction (blue) but no actual insulin in injected (red).



(A) coloring based on true nature (B) coloring based on LOF anomaly score (C) coloring based on iForest anomaly score

FIGURE 5.5: Simulated patient #001, 3D scatter-plot of $\phi_f(t) = [g(t),\ DCOB(t),\ ICOB(t)]$.

Unluckily, hyperglycemic episodes associated to prolonged insulin suspension can not be recreated a-posteriori. Instead, unexplained hyperglycemia episodes were emulated by removing from the data a meal and removing also the associated insulin bolus.

A more solid real-data analysis requires the design of complex ad-hoc experiments, such as those performed in (Cescon et al., 2016; Howsmon et al., 2017), and has to be deferred to future extensive clinical validations.

5.4 Algorithms Tuning and Testing on synthetic data

5.4.1 Evaluation Criteria

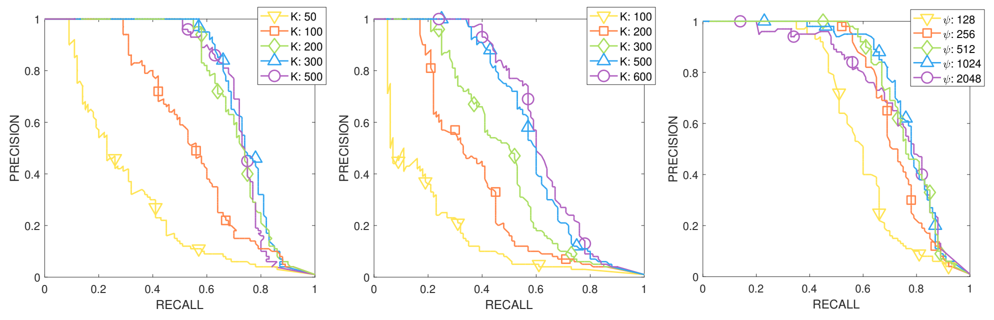
While assessing the performance of the algorithms, it is crucial to account for the delayed effect of insulin on glucose, that makes it impossible to immediately detect a pump fault. Instead, we are interested in detecting a fault within $T_{\max}$ from the event start. Hence, we define a True Positive (TP) event when at least one alarm is produced within at most $T_{\max}$ from when the fault begins and a False Negative (FN) event when a fault occurs, but no alarm is raised in the following time-window of length $T_{\max}$. A False Positive (FP) event occurs when an alarm is produced, but no fault occurred in the previous time-window of length $T_{\max}$. Furthermore, given that the system remains in an anomalous state for some time after the restoration of the pump normal functioning, we suspend the evaluation for a time-window of length T_{sus} after the end of the fault and possible alarms occurring in that time slot are not labeled as FP. Reporting True Negatives (TN) for this problem is complex and of limited interest, given that we are dealing, as in real-world scenarios, with highly unbalanced dataset. For such reason, TNs are not reported here.

Beside TP, FN and FP events count, we also consider the standard metrics Recall (REC) $= \frac{\text{TP}}{\text{TP+FN}}$; Precision (PR) $= \frac{\text{TP}}{\text{TP+FP}}$; F1-score (F1) $= 2\frac{\text{PR}\cdot\text{REC}}{\text{PR+REC}}$.

$T_{\max}$ is set to $T_{\max} = 6$ hours, since after this time the impact of the fault on blood glucose is expected to be apparent even to the patients, making alarming superfluous. Similarly, we define $T_{\text{sus}} = 6$ hour, as this represents the time needed to re-establish a non-anomalous physiological condition in the patient after the termination of the fault and the restoration of the normal functioning of the pump.

First of all, we illustrate in Fig. 5.5 the results produced by the algorithms by depicting the anomaly score they assign to each point of the dataset, on a case study subject of the simulator. Inliers are concentrated in a medium-high density cloud, while outliers belongs to the tail-shaped region. We observe a difference in the behavior of LOF and iF, since iForest tends to assign a larger anomaly score at the borders of the main points cluster with respect to LOF.

5.4.2 In-silico Tuning



(A) LOF (B) COF (C) iF

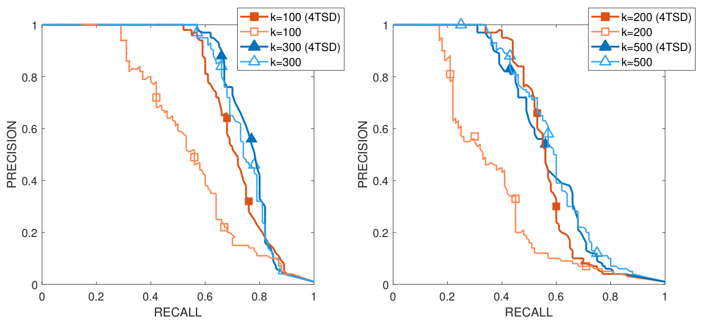
FIGURE 5.6: PRC analysis for threshold selection and parameters tuning of the methods. Results reported for LOF (a), COF (b) and iForest (c). Different points in each PRC curve corresponds to different threshold value for respective anomaly scores.

We tune each of the proposed algorithm analyzing the Precision-Recall Curve (PRC), the better analysis tool when dealing with imbalanced datasets. (Saito and Rehmsmeier, 2015). The results are shown in Fig. 5.6a, Fig. 5.6b and Fig. 5.6c for LOF, COF and iForest, respectively. Each method has specific tuning parameters: LOF and COF require to choose a suitable value of k (the neighborhood cardinality); iForest requires to chose the parameter ψ (subsample size), while we kept fixed the number of trees $T = 100$, as it proved to be of little impact on the method performance. For each value of this tuning parameter, various values of the threshold τ^0 on the anomaly score are considered, drawing a curve in the PRC.

Fig. 5.6a shows that as k increases from $k = 50$ to $k = 300$, the curves move towards the top-right corner, meaning simultaneous increase of both precision and recall, at the expenses of an increased computational cost. For larger values of k, no significant improvement is observed, since k has become big enough to make the k-neighborhood representative of the entire dataset. COF (Fig. 5.6b) presents a similar response when increasing k, but a worse overall performance w.r.t. LOF. This is most likely due to the fact that COF is designed to consider *connected* structures (such as tail-shaped) as normal instances, hence resulting in lower score in anomaly regions of the problem under investigation. Fig. 5.6c shows that, for the iForest, performance increases up to medium sub-sample sizes ($\psi = 512$), then stabilizes ($\psi = 1024$) and even decreases when ($\psi = 2048$). This is due to the fact that when the sample size is too small, iForest sub-sampling is not able to capture the real distribution of data, while, when the sample size is too large, anomalies appear too often in the sub-sample causing the, so-called, masking effect mentioned before. Moreover, we recall that higher k and ψ result in linearly higher computation costs.

5.4.3 Impact of 4TSD Adaptation & Root Cause Analysis

Fig. 5.7 investigates the effect of 4TSD adaptation. The parameters T^P and T^F were manually tuned and set to $T^P = 180$ min and $T^F = 360$ min. When using a very low value of k, LOF-4TSD and COF-4TSD are able to reach comparable performance to normal LOF/COF with a higher k (≈ 3 times). This means that it is possible to



(A) LOF (B) COF

FIGURE 5.7: Impact of the 4TSD adaptation on the detection performances on LOF (a) and COF (b). Different points in the PRC curve corresponds to different threshold values for respective anomaly scores.

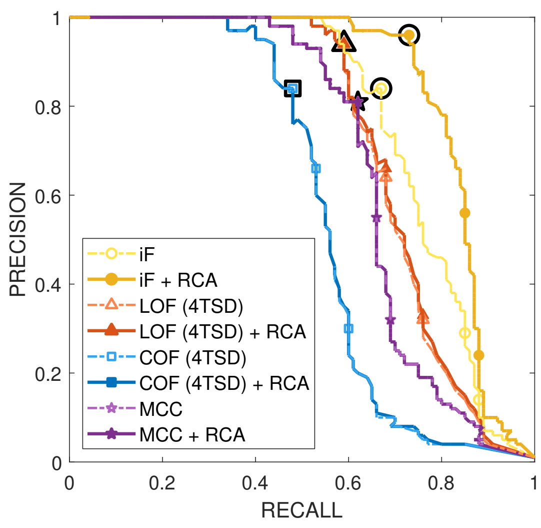


FIGURE 5.8: Comparison of the four AD methods, with and without RCA adjustment. Different points in the PRC curves corresponds to different threshold values for the respective anomaly scores. For each curve, the best threshold settings in terms of F1 score is highlighted with a black marker.

maintain the same AD performances while remarkably decreasing the computational burden. On the other hand, for higher values of k, the effects of 4TSD vanishes.

Based on the above analysis, for each algorithm we chose the hyperparameter settings that granted the best trade-off between computational costs and performance: for LOF and COF this was obtained with the use of the 4TSD adaptation and by setting $k = 100$ and $k = 200$ respectively; for iForest, this was obtained with $\psi = 512$.

Setting these optimal hyperparameters values, we tested the computational time of the presented algorithms. In Table 5.2, we report the results obtained on a desktop PC.[5], when considering a computationally demanding algorithm implementation that, whenever a new data is read, re-evaluate the anomaly score on all the previously collected points (30 days of simulated data). While this will likely be overoptimistic with respect to the implementation of the same algorithm on a wearable AP hardware, the proposed estimate is very conservative with respect to case of implementation on a remote server. It should also be taken into account that the code was Matlab based and

[5]Specifications: Windows 10, Intel(R) Core(TM) i7-7700K CPU @ 4.20 GHz 4.20 GHz, RAM 16 GB, simulation performed in MATLAB

not optimized, e.g. with a dedicated C implementation. Finally, a substantial further reduction of the computational time can be achieved by resorting to implementations of the algorithms optimized for on-line use (Ding and Fei, 2013) (Goldstein, 2012).

Computational times obtained are remarkably shorter than the application sampling time (5 minutes) in the case of LOF and iForest. In the case of COF is sufficiently shorter than the application time but better analysis are needed to ensure its portability.

Regarding the choice of the anomaly score threshold value, τ^0, we chose the value that maximized the F1 score. This criterion gives the same importance to both precision and recall. Other criteria could be used in cases in which recall is more or less important than precision; for instance, in a remote monitoring context, FP could be considered less critical than FN given that human experts will monitor the AP systems and analyze the alerts.

As anticipated in the Methods section, the anomalous data portion detected by the algorithms might simply correspond to rare events (most relevantly hypoglycemia) and not necessarily to events associated to insulin pump faults. For this reason, based on the observations performed with RCA, we introduced a simple heuristic, specifically aimed to distinguish between pump faults and hypoglycemia. We obtained this by impeding the algorithms to produce an alarm when a concomitant $g(t)$ value lower than $g_{\text{th}} = 120 mg/dl$ was observed. We will refer to the usage of this strategy as "RCA" from here on.

The impact of RCA in each AD technique is shown in Fig. 5.8, where, for each algorithm, we report the PRC obtained by varying the anomaly score threshold to τ^{RCA}, before the application of RCA (dashed line) and after (continuous line). Each algorithm was used considering its optimal setting, previously identified. Fig. 5.8 shows that only iForest benefits from this procedure, pointing out that this algorithm is the one that detected most anomalies not associated to faults. This is supported by comparing Fig. 5.5b and Fig. 5.5c where that iForest tends to assign a larger anomaly score at the borders of the main points cluster with respect to LOF, thus being more inclined to produce FPs.

Finally, in Fig. 5.8 we highlighted with a black marker the point in the precision-recall space with the highest F1 score value; the threshold s_{th} associated with that point will be considered as the optimal threshold value for the AD method and will be used in the following.

5.4.4 Algorithm Comparison and in-silico testing

This section reports a comparison between the AD algorithms presented and a traditional multivariate control chart (MCC) approach, used as a baseline approach. Specifically, using the same feature set described before, we fitted a multivariate Gaussian distribution to all data-points $x(t) = [g(t), DCOB(t), ICOB(t)] \in \mathbb{R}^3$ of a patient and we evaluated the cumulative distributive function at each data point, using its percentile as the anomaly score. Alarms are issued when the anomaly score

TABLE 5.2: Time needed for anomaly score computation, 30 days of data on a desktop PC.

Algorithm	Time (s)
iForest	11.43 ± 0.19
LOF (4TSD)	5.58 ± 0.08
COF (4TSD)	35.70 ± 0.79

surpassed a threshold tuned with the same rationale used in the previous section (Fig. 5.8).

The methods are tested on a new 30-day dataset (test-set), completely disjoint from the dataset used to tune the methods parameters (training-set). This test-set is obtained using the same physiologic simulator and simulation scenario described in Sec. 5.3 and different random extraction of all the random parameters.

Methods performance in these configurations are reported in Table 5.3. At the optimal s_{th} and before RCA application, similar performances were observed between iForest and LOF: the recall is ∼60% and precision ∼95%. COF performance was inferior: ∼50% recall with ∼75% precision. Applying the proposed RCA rules to iForest allows to reduce the number of FPs, hence decreasing the value of s_{th} that maximizes F1. In this new configuration both sensitivity and precision are increased: 81% recall, together with ∼92% precision, outperforming LOF.

In Table 5.4, we report the distribution among the simulated subjects of false positives occurred in the test-set, obtained by each algorithm with the optimal setting choice. We can observe that the distribution of FPs is distributed among the virtual subjects. This excludes the possibility of the statistics being unfairly biased by hypothetical "bad" subjects.

Finally, we investigated the possible association of FPs and the patients error in computing the pre-meal bolus. In Fig. 5.9, we report a box-plot of the error estimating the carbohydrate content (e_{CC}) of the meal, that in turn results into an error in the calculation of the associated bolus. A positive value represents an overestimation of carbs, hence an overdosing of insulin. A negative value represents an underestimation of carbs and hence an underbolus, likely to result in hyperglycemia. Of note, this information is known only in simulated data. Over imposed to the box-plot, a scatter plot of the errors, depicted as big red diamonds when followed by a FP committed by the algorithm or by a small gray circles when not followed by a FP. In the case of iForest, it can be noticed that there are numerous underestimation errors with absolute magnitude similar or larger than those that were followed by a FP. This is reassuring on the functioning of the algorithm, since it proves that this method is not systematically fooled by large under-boluses and the resulting hyperglycemia is not systematically confused with a pump fault. On the other hand, the fact that 6 out 7 FP occurred after an underestimation error supports the very reasonable conjecture, suggested by the anonymous reviewer, that bolus underestimation and the resulting unexplained hyperglycemia represent a major source of error for fault detection methods and one of the key challenge in this problem. LOF data seem to suggest analogous considerations, but there is no sufficient statistical evidence to draw solid conclusions.

In the case of COF, the results show that carb-counting underestimations are

TABLE 5.3: Comparison of algorithms performance on the test-set

	LOF		COF		iForest		MCC	
	τ^0	τ^{RCA}	τ^0	τ^{RCA}	τ^0	τ^{RCA}	τ^0	τ^{RCA}
TP	58	58	50	50	65	**81**	60	60
FP	4	4	14	14	5	**7**	13	13
FN	42	42	50	50	35	**19**	40	40
REC [%]	58	58	50	50	65	**81**	60	60
PREC [%]	94	94	78	78	93	**92**	82	82
F1 [%]	72	72	61	61	76	**86**	69	69

TABLE 5.4: False positives (FP) per patient

	LOF		COF		iForest		MCC	
# FP	τ^0	τ^{RCA}	τ^0	τ^{RCA}	τ^0	τ^{RCA}	τ^0	τ^{RCA}
1	4	4	7	7	5	5	1	1
2	0	0	2	2	0	1	1	1
3	0	0	1	1	0	0	0	0
4	0	0	0	0	0	0	1	1
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	1	1
>6	0	0	0	0	0	0	0	0

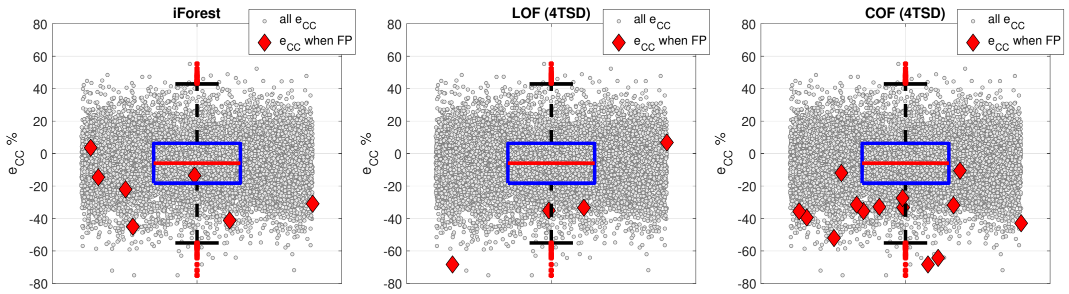


FIGURE 5.9: Association between carb-counting error e_{CC} and the possibility of FP. Scatter plot data point are randomly located on the x axis to help the visualization.

a major cause of FP for the algorithm, since 13 out of 14 FPs are associated with underestimations, and 11 out of 14 FPs are associated with underestimations below the 75th percentile.

Finally, in Fig. 5.10 we report the boxplots of detection time distribution of our algorithms, showing a median of ∼4 hours from the fault start for all three methods. It is important to remark that these results are obtained knowing the exact fault start, which is possible only in simulation, while in a real dataset the detection time can only be computed based on the first visible impact on blood-glucose, occurring ∼2h after the pump stops delivering insulin (see Fig. 5.4a). Therefore, the observed median detection time of ∼4 h from the "true" fault start (known on only in simulation) corresponds to ∼2 h from the first visible effect on blood-glucose.

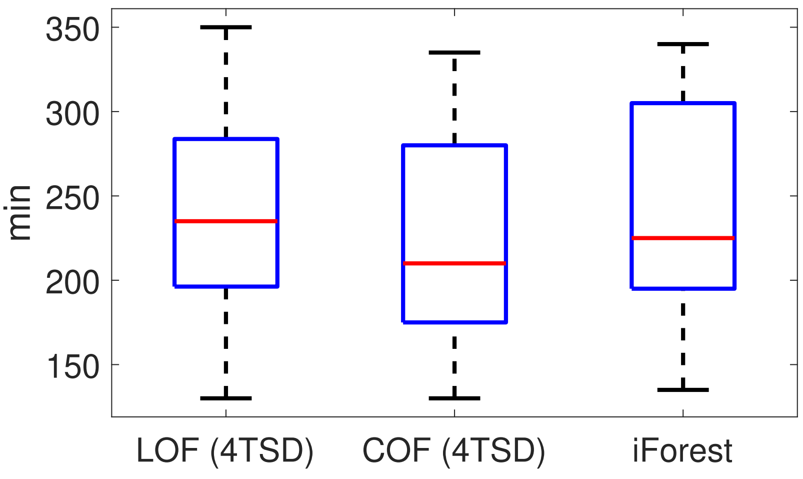


FIGURE 5.10: Boxplot of detection time obtained with LOF, COF and iForest

In conclusion, the simulated-data analysis empirically showed iForest as the most effective method for our problem. In the following section we will therefore focus on this algorithm and the baseline method MCC.

5.5 Real-data Testing

Fig. 5.11 shows the data of patient #06 in the feature space. Each data point is colored according to the anomaly score assigned by iForest. A visual comparison with the synthetic data of Fig. 5.5c shows that, in real-data, is still present a well clustered cloud of non-anomalous points but the deviation from such cluster are more frequent than in simulated data, due to the higher complexity and larger uncertainties.

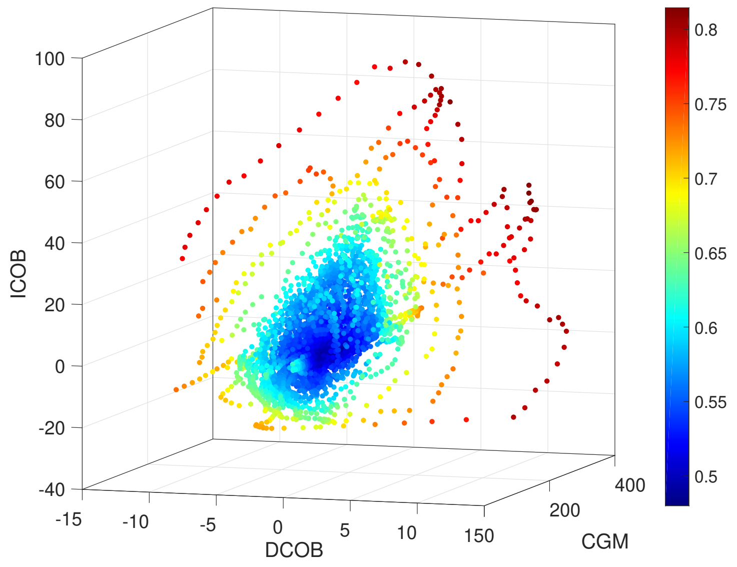


FIGURE 5.11: Scatter-plot of data of patient #06, coloring based on iForest anomaly score

As for the retrospectively labeled dataset, the domain-expert team identified 6 "Fault" episodes , i.e. anomalous hyperglycemic episodes caused by a pump fault or other issues producing similar effects, 3 "Other" episodes, i.e. anomaly not due to a pump malfunctioning and 29 "Unclear" events, i.e. suspicious data portions lacking of sufficient evidence pointing out a fault. That amounts to < 1 "Fault" and "Other" episodes per patient per month and ~ 4 "Unclear" episodes per patient per month.

As before, we define a TP event when at least one alarm is produced within at most $T_{\max} = 6h$ from when the fault begins and a FN event when a fault occurs, but no alarm is raised in the following time-window of length $T_{\max} = 6h$. On this dataset two types of false positives were considered: the standard FP, occurring when an alarm is produced but no "Fault" episode occurred, and the certain "False Positive (cFP)", in which we count only false alarms which did not occur within $T_{\max}$ form an "Unclear" episode. Apparently, cFP $\leqslant$ FP.

Table 5.5 reports the performance of iForest vs MCC in each of the seven patients. At the population level, iForest is able to detect 5 "Fault" episodes out of the 6 identified in the dataset. This high recall is maintained with low FP rate, in fact 8 episodes were incorrectly labeled as faulty and 3 of these 8 were considered suspicious also by the human operators. This amount to an average of 1 FP per patient in a month of use. A total of 8 episodes were incorrectly labeled as faulty and 3 of these 8

TABLE 5.5: Performance on retrospectively labeled dataset

	Label		iForest				MCC			
Subject	Faults	Unclear	TP	FN	FP	cFP	TP	FN	FP	cFP
Sub01	0	4	0	0	0	0	0	0	6	6
Sub02	2	5	2	0	1	0	2	0	6	4
Sub03	0	6	0	0	2	0	0	0	6	4
Sub04	0	5	0	0	1	1	0	0	9	9
Sub05	1	4	1	0	0	0	1	0	8	6
Sub06	2	2	2	0	2	2	2	0	4	3
Sub07	1	3	0	1	2	2	0	1	8	7
Total	6	29	5	1	8	5	5	1	48	33

were considered suspicious also by the human operators. Considering the length of the dataset, the obtained incidence of FPs is very satisfying in practice: in average only 1 FP per patient per month, that seems highly acceptable in practice. iForest clearly outperforms MCC that has similar recall but drastically higher incidence of FP, incorrectly labeling 48 events ($\sim$ 7 per patient) out of which only 15 were considered suspicious by the operators (cFP= 33).

Finally, we consider the dataset with faults introduced a-posteriori. In this dataset, data portions labeled as "Fault" and "Unclear" by the operator were removed, hence FP always equals cFP. One fault per patient was included by canceling a meal (and the associated bolus). Table 5.6 shows the detection performance the methods at a patient level. At the population level, iForest shows good recall, detecting 6 faults out of the 7 introduced a posteriori. Equal recall is obtained by MCC. Table 5.6 also reports the number of FP recorded since, due to the dataset changes, the number of FP is expected to change form the previous case. In fact, iForest commits only 2 FP largely outperforming MCC, that commits 37 FP.

As a concluding remark, it should be noted that these performances were obtained without retuning the parameters of the algorithms, i.e. the parameters ψ and τ^{RCA} of iForest and the parameters τ^{RCA} of MCC used in this section are those previously tuned on the simulator.

TABLE 5.6: Performance on a-posteriori modified dataset

	iForest		MCC	
	Detected?	FP	Detected?	FP
Sub01	YES	0	YES	6
Sub02	YES	0	YES	6
Sub03	YES	0	YES	5
Sub04	YES	0	YES	9
Sub05	YES	0	YES	4
Sub06	NO	2	NO	2
Sub07	YES	0	YES	5
Total	6/7	2	6/7	37

5.6 Future directions

In this thisis, we first proposed a (low-dimensional) feature set capable to account for the dynamics in T1D physiology and then, via a nonlinear transformation, modified it into a feature set capable to highlight anomalous behaviors associated to pump faults. Further feature crafting, possibly including additional signals such as physical activity or delivery of other hormones/drugs, and extensive feature engineering, albeit appealing for its potential impact on the detection performances of the method, goes beyond the scope of this work and it is differed to future investigation. In this respect, we should also point out that the employed techniques, especially iForest, are able to cope with medium-high feature dimensions p (e.g., $p = 100$): this flexibility allows to easily add new features and then increasing the performance without incurring in computational issues.

Furthermore, in this work we assumed the availability of 30 days of patient's historical data. This could be an issue at the start-up on a new patient. However, in practical implementation (both embedded in the AP or in Internet-of-Things implementations) AD methods could exploit pre-loaded population data, instead of patient data.

We also remark that here we used a unique pre-computed threshold for all patients. However personalized thresholds can be envisioned for enhanced accuracy.

Additionally, the present work focused on insulin pump faults, however, the method proposed can be extended to the detection of anomalies caused by other malfunctioning such as sensor faults or communication errors and to detect incorrect patient-provided information. Covering these cases is of major interest and will be investigated in future works.

Finally, an interesting future work is to investigate the performance improvement achievable on labeled datasets potentially available by resorting to supervised or semisupervised methods.

5.7 Comments

The most effective of the proposed methods, iForest, underwent a proof-of-concept validation on real-data of seven patients that have used an AP system for 1 month in free-living condition. This dataset lacked of labels on the functioning/faulty status of the pump, so it was retrospectively labeled by a team of domain experts. Since this procedure is prone to errors and uncertainties, we also consider a modified a-posteriori dataset, where only certainly not-faulty data were retained and where a fault per patient was introduced a-posteriori tampering with recorded data and canceling a meal (and the associated meal-bolus). Albeit affected by several limitations, this proof-of-concept validation on real-data complements the in-silico analysis proposed and shows very encouraging performance. Notably, these performances were obtained without need of retuning the algorithm.

This results pave the way for more extensive and solid testing on real-data. The presented anomaly detection approaches can be used embedded locally on AP systems but they can also be deployed in remote health-care monitoring systems (Arsand, Varmedal, and Hartvigsen, 2007; Islam et al., 2015), by exploiting the potential of an AP system, one of the medical devices more suitable to be used as Internet-of-Things device. The introduced method expands the landscape of existing fault detection strategies, with the possibility to construct more sophisticated fault detection solutions that exploit multiple fault detection techniques in parallel.

Chapter 6

Conclusion and Perspectives

Time series-classification (here we also include anomaly detection) is a broad field of research that demands algorithms tailored to the specific application.

An important factor to bear in mind is the trade-off between decentralized and centralized systems. In decentralized systems, one has to employ cheap algorithms that do not require high computation and memory capability. In this part of thesis, we focused on such algorithms in the context of AR/GR and AP.

Among the issues that characterize these applications, the following have been addressed:

- **Parsimony**. Wearable devices are generally resource constrained. This calls for parsimonious but effective algorithms. We used sparse representations that either use few support vectors or a symbolic description that allow to easily signals with a ridiculous amount of memory and computations, keeping models interpretable and cheap.

- **Invariance**. There are many factors to which algorithms have to be invariant. In this context, we addressed invariance to rotation and time warping.

- **Data heterogeneity**. Data come from many users in different forms and thus models have to be flexible enough. We provided models that are effective also with new users.

- **Time-correlation**. Regarding AP, we introduced a simple technique that allows to apply ”static” anomaly detection approaches to time-series data.

The second part will be devoted to deep learning approaches that normally run on the centralized system. In such framework, there are few constraints on memory and computational power. However, as we will see, other issues arise.

In the following, we will present future perspectives of the research presented in this part. In Sec. 6.1 we present a preliminary idea about how to ameliorate current limitations of the SAX method. Sec. 6.2 is devoted to present future perspectives of anomaly detection for AP using deep learning methodologies.

6.1 Action/Gesture recognition trough elementary actions

Despite its effectiveness, SAX is not optimal from an information-theory point of view, because the procedure to create symbols is poorly data-driven. Moreover, contiguous symbols are created independently, losing the information given by the dynamics.

How can we create data-driven alphabets? The first problem to be solved is how to segment the signal. Then, once a segmentation method has been chosen, symbols are built through canonization, that is normalization of signals; however, in most of real scenarios and specifically in the problem at hand, normalization approaches (both in time and amplitude) may not yield satisfying results because the nuisance factors act in a such a way that signals representing *unique* gestures are not trivially "deformed". Fig. 6.1 depicts this fact: the "true" alignment is not well defined since deformation of real signals does not follow linear affine transformations.

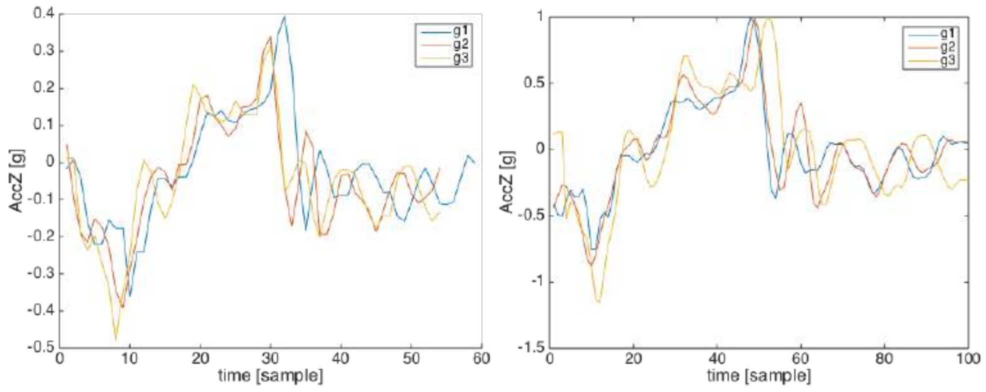


FIGURE 6.1: Atomic gesture normalization. Left figure represents the original atomic gestures: they have different time and amplitude scales. Moreover, these deformations are not linear since are they change in different portion of the gestures. Right figure represents the canonized gesture: blue signal was taken as reference gesture and the amplitude normalization factor is the maximum amplitude w.r.t. the reference signal amplitude. Time scale was normalized using resampling and translation was evaluated using the pontifical feature. The canonization is not able to align the gestures: the deformations are not of the kind $at + b$.

As we can see that sub-signals are deformed in different ways yielding global non-linear deformations. On the other hand, this give us the intuition the each atomic gestures may be thought as a sequence of sub-gestures or actions. In fact, most activities can be decomposed in *gestures* that may be performed with different speeds and power, reflecting local time and space deformations on sensor's signals. The first problem to face is how to perform the decomposition, that is, how to recognize the complex actions in a gesture: we can think actions, in turn, composed by elementary actions. Thus, each atomic gesture is a sequence of elementary actions. These actions can be retrieved using the so-called border features, that is the ones that define how to individuate the delimiters of elementary actions. The simplest ones are local extrema. A natural way to define an atomic actions is the same with which we define an atomic gestures, i.e., the trajectory around a local extrema. In fact, there are several procedures to define elementary actions, among which the four simplest ones are:

1. representing these as fixed-length segments of $K + 1$ sample around the extremum (i.e. $K/2$ towards left and $K/2$ toward right),
2. representing these as the segments with borders left and right extremum and center the current extremum,
3. representing these as the segments growing symmetrically from the center extremum and up to the first encountered extremum,
4. representing these as the segments between to local extrema.

Method 1 is simple but the fixed length means that is not able to capture both slow and fast dynamics with the same effectiveness. Method 2 has the advantage to be adaptive to variations of activity frequencies but is not symmetrically distributed around the center. Methods 3 is adaptive and symmetrically distributed around the center. Methods 4 may have the advantage to be more (locally) invariant to deformation but it may yields in a set too fragmented and uniformative signals.

The crucial idea of this work consists on forming a canonical basis $\{a_1, \ldots, a_k\}_{k=1}^n$ of elementary actions, such that each atomic gesture is an ordered sequence $a_{i_1}, \ldots, a_{i_k}$, $i_j \in \{1, \ldots, k\}$. In forming this basis, we need to define a distance between elementary signals: as claimed above, the same action can be performed with different time and space scales and then euclidean distances are not directly applicable and DTW-like distances may be used. The canonization step aims to normalize time and space scale in order to make signals comparable with a simple L1-distance. Once defined a distance, a basis may be formed by clustering K different elementary signals. In recognising complex activities, that may be similar, temporal e spatial information could be crucial to distinguish gestures: in this case, reducing, apparently same signals, on the same time or space scale may discard useful information. For example, walking and running could be comparable once normalized to the same scale. Thus spatial and temporal information must be accounted for when modelling a recognition model. Intuitively, different persons have their own signature on performing a gesture but, normally, they act similarly, i.e., temporal and spatial scale should not be very different. Therefore, we may assume that two signals similar in the shapes but very different on time and space scales represent two distinct activities.

Thus, under this rationale, we can think time and space domains as clustered in K_t time and K_s space scales, respectively. Each 2d-scale (time and space, jointly) admits a certain number S of *shapes* that distinguish different gestures or actions. Thus the canonical basis shall be composed by $K = K_t \cdot K_s \cdot S$ signals. Another feature that, at least in general, has to be considered is the *mean*: different means scales may represent distinct gestures. The simplest way to consider this feature is to quantize it in K_m bins analogously to time scales. This yields to $K = K_m \cdot K_s \cdot K_t \cdot S$. A more efficient way to consider the feature mean is to form the canonical basis with zero-mean segments and exploit this information directly in the classification phase adding it as feature. This simplification has as by-product that when the feature *mean* is uninformative can be automatically discarded by the classifier. However, even with this simplification, K may easily reach order 100 (e.g. $K = 5 \cdot 5 \cdot 5$) resulting too memory-expensive for wearable devices. In order to reduce the complexity of models we could:

1. considering the space scale analogously to the mean quantization, i.e. subdividing the space scale in K_s bins and considering this as an additional feature. This method has the great advantage that the number K reduces by a factor K_s;

2. encapsulating space scale information in the S different shapes for each time scale, i.e. setting up S such that enough space information is extracted.

All the gestures can be represented as strings (sequence of elementary actions): thus, the gesture recognition problem is essentially a string matching problem, in which similarity between string can be evaluated from string kernels. Then, the last step consists on the classification phase using SVM or RVM classifier. In the following we describe the algorithm procedures more in detail.

6.1.1 Canonization

First extract all the detrended sub-segments appertaining to the training set representing the elementary actions, then quantize in K_t equal intervals and define a canonical length T_i: this could be chosen simply selecting a particular sub-segment as canonical (specially easy with off-line learning) or imposing $T_i' := \frac{T_{max}}{K_t}(i - 0.5)$ where T_{max} is the longest sub-segments' length. A more conscious choice is $T_i := \frac{\sum_{i=1}^{K_t} c_i T_i'}{\sum_i c_i}$ where c_i is the number of sub-segment laying in interval i; T_i is nothing else that the centroid of cluster i. Then quantize the means of sub-segments in K_m bins and space domain in K_s bins (otherwise follow method (b)).

Once formed the basis of K elementary signals, when a new sub-segment is detected it is *locally*[1] normalized in time and space domains as described above.

There are two main shortcomings:

1. The parameters K_t, K_m, K_s (if any) and S must be chosen a priori. Often, prior knowledge is needed in order to select optimal parameters. On the other hand, other clustering techniques as hierarchical clustering (Hastie, Tibshirani, and Friedman, 2009) do not need a choice of the number of cluster a priori. However, we prefer K-means due to its low time-complexity;

2. the number of shapes S is equally fixed for all time scales: this may lead to a loss of information on one side and redundancy on the other side. In fact, some time scale could be dense of information, i.e. with a dense number of distinct shapes, and, by contrast, other time scales could be over-represented.

6.1.2 Classification

The above procedure yields the basis $\{a_1, \ldots, a_K\}$ of the elementary signals. Then, all gestures will be projected into this basis space resulting in a collection $\mathcal{C} = \{s_i, \mid i = 1, \ldots, N\}$ of strings $s = \{a_{j_1} \cdots a_{j_n}\}$, $|s| = n$, where N is the number of training predictors, n is the sequence length and $i \in I_K = \{1, \ldots, K\}$. The simplest way to perform the classification task is to count the frequency of elementary actions in each gesture. The resulting predictor is a vector of K frequency counts. Thus the resulting design matrix D is $N \times K$ where K is the cardinality of the canonical basis. This method does not capture the temporal information, i.e. it does not induce an order constraint: for example string *aab* and *aba* are treated as equivalent albeit they may represent two different gestures. Temporal information can be exploited using string kernels vastly used for protein classification (Leslie and Kuang, 2004) and a standard classifier (such as SVM).

6.1.3 Elementary Actions

Here we shall describe the procedure that only considers time-scale canonization and not amplitude-scale: the method is easily extendable. For the sake of simplicity, we assume to use the K-means algorithm for clustering.

First, extract all the sub-segments around local extrema and group all their duration and run K_t-means clustering, i.e. quantize in K_t representative time scales. Then, resample retrieved signals to the nearest time scale. This permits to avoid using DTW techniques given that we compare signals over the same time scale. Here the crucial idea is to be *locally* invariant to small deformations but not to large time deformations. In some application, this rationale is not applicable and a unique time-scale should be considered.

Different gestures or actions present different shapes: thus for each time-scale, K_s different elementary actions will be considered. The total number of elementary

[1]The term locally means that the signal is not totally invariant but it is invariant only for small deformations. Remember that this properties is consequence of considering signals at remarkable different scales as representing distinct signals even though shapes are similar.

actions is $K = K_t \cdot K_s$. In order to identify this basis, for each time-scale a K_s-means clustering is run using the L1-distance. The input matrix is $N_t \times p_t$ where N_t is the number of training signals at the given scale and p_t is the number of samples of signals at the given scale.

Some particular actions may present equal shapes but different means, that is, the mean can be a distinctive feature to identify gestures. On the other hand, in some other activities, the mean is uninformative. In fact, an athlete performing a gesture in inclined planes yields different accelerations than one in a flat plane. Here we suppose the mean is informative: if not, it is sufficient not to consider it. Each basis signal s_i can be represented by a symbol a_i and then the basis is $\{a_1, \dots, a_K\}$. Each gesture can be represented by a variable-length sequence of elementary actions. The result of the procedure presented above is a collection $\mathcal{C} = \{x_i, | \ i = 1, \dots, N\}$ of sequences $x = \{a_{j_1} \cdots a_{j_n}\}$, $|x| = n$, where $N = \sum_{t=1}^{K_t} N_t$ is the number of training predictors, n is the sequence length (not fixed) and $j \in I_K = \{1, \dots, K\}$.

The canonization procedure can be tailored to the specific application. For instance, in gesture recognition, we may be more interested in shapes rather than amplitude or time scales or both.

6.2 Future works for AP

As a future work we plan to tackle the anomaly detection for AP using Deep Learning techniques. As we already showed in Chapter 5, one of the standard approaches is to fit a physiological model using Kalman filters and then detect the anomaly by means of statistical tests on the prediction residuals. However, real physiological systems are very complex and very variable (e.g. child vs adult). And usually, these models fit well an average trend, but fail in capturing peculiar signatures, making the test of residuals ineffective and unreliable.

In the context of Deep Learning, there are two lines that can be followed:

- using common generative frameworks, such GANs (Sec. 7.4) with the task of fitting CGM data conditional to input signals
- solve an unsupervised problem as a supervised classification. For example, instead of trying to match and predict the CGM signal, we could set artificial classes on future windows such as "decreasing/increasing", "derivate more/less than" etc. This approach has two advantages: the first is that training a supervised model is much easier than training a GAN and the second is that we can easily design classes by exploiting domain expert knowledge.

We will explore both the directions.

Part II

Robust and interpretable Deep Representations

Chapter 7

Motivation and background

In the last years, with the emergence of large datasets, such as (Deng et al., 2009), Deep Learning (DL) led to a revolution in many different research areas, such as Computer Vision (Krizhevsky, Sutskever, and Hinton, 2012), Natural Language Processing (Young et al., 2018), Speech Recognition (Nassif et al., 2019) and Healthcare (Esteva et al., 2019). State-of-the-art results of deep learning on these fields, overwhelmed fifty years of careful expert feature engineering. However, the no free lunch theorem holds: we always have a price to pay for. Indeed, although results for DL are impressive, it is not trivial to 'open' the black-box. Especially in applications where humans are involved, robust and interpretable models are paramount. Unfortunately, as we will show, common state-of-the-art DL models do not share these two key properties. This fact motivated recent research directions (Alvarez-Melis and Jaakkola, 2018; Madry et al., 2017) and the following two chapters. In Chapter 8 we will review recent results on robustness of DL and we will propose an algorithm which provides a tool to control the accuracy-robustness trade-off, given a cost that is tailored for the application at hand. Chapter 10 is devoted to the definition of intepretable classification models. Moreover, we will establish a connection between transportation theory, robustness and interpretability. Before continuing, we would like to focus on the definition of interpretabilility: in the literature it is often vacuous and not precisely stated. In this work, we informally say that a model is interpretable if its output aligns with a human point-of-view. Thus, in principle, we are not interested in what happens inside the model but only in its input-output mapping. We will formalize this concept later on.

7.1 Notation and background

The objective of this section is to fix the notation used in the following two chapters and to provide the minimum but necessary background in order to produce a self-contained elaborate to facilitate the reader.

7.1.1 Artificial Neural Networks

Artificial Neural Networks are the foundation of DL technologies. An ANN is the interconnection of simple units called *neurons* in a multilayer structure that emulates a simple brain. We can distinguish three different types of network layers: input, hidden and output. The input layer provides the input values to the network. The output layer provides the output of the network and its structure is chosen in order to match the characteristics of the output; in particular, a *regression* function is employed when the output is continuous i.e. $y \in \mathbb{R}$, while a *classification* function (e.g. softmax) is used (Friedman, Hastie, and Tibshirani, 2001) when the output is categorical which means $t \in \{0, 1, \cdots, K\}$ where K is the number of classes; hidden layers apply a

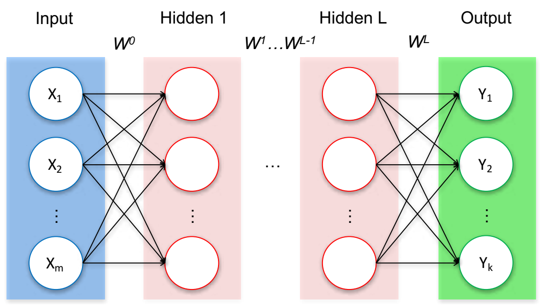


FIGURE 7.1: Generic structure of a FNN. It is possible to distinguish the input layer (blue) output layer (green) and L hidden layers (red).

transformation to the previous layer's output, the transformation depends on the structure of the hidden layer itself. Various ANN structures have been developed over the years; the simplest one is the so-called Feedforward Neural Network (FNN) (Fig. 7.1) where the neurons are connected in a directed graph without any feedback loops. In this case, the hidden neurons apply a non-linear activation function σ to an affine transformation of the previous layer's output. We can thus associate a matrix W^l and a bias vector b^l to each hidden layer l whose output can be computed as follows:

$$y^l = \sigma(W^l y^{l-1} + b^l) \tag{7.1}$$

here $\sigma(\cdot)$ denotes element-wise application of the activation function σ. Given the number of neurons at l^{th} layer q^l, and the number of neurons at layer $l-1$ q^{l-1}, the matrix W^l has size $q^l \times q^{l-1}$ and the bias vector b has length q^l. The output of the $(l-1)^{th}$ layer is a column vector of dimension q^{l-1}.

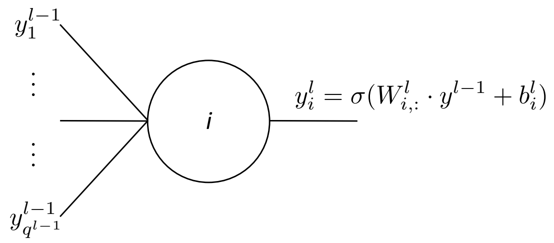


FIGURE 7.2: Output of the i^{th} neuron of the k^{th} layer of a FNN. The notation $W_{i,:}^l$ indicates the i^{th} row of the matrix W^l

Common choices for activation functions σ are the *sigmoid*, *tanh* and *rectifier linear unit (ReLU)* (Nair and Hinton, 2010). Usually, the *ReLU* function is a good choice because of its similarity with a linear function and thanks to its consistent gradient that does not vanishes during training (Goodfellow, Bengio, and Courville, 2016b). The ReLU function is defined as:

$$\sigma(x) = \max\{0, x\} \tag{7.2}$$

In recent years, more complex networks called Convolutional Neural Networks (CNNs) have gained more and more popularity thanks to the advancements obtained in Computer Vision. CNNs, exploit a multilayer structure similar to FNNs but with different types of hidden layers, which are alternated. In particular we can distinguish three kind of hidden layers: (i) convolutional, (ii) pooling, (iii) fully connected.

(i) Convolutional layers are similar to the one employed in FNNs but in this case, each neuron applies the activation function to the convolution of the previous layer's output with a kernel W^l plus a bias term b^l. The output of the l^th convolutional layer can then be computed as follows:

$$y^l = \sigma(W^l * y^{l-1} + b^l) \tag{7.3}$$

Usually, multiple kernels are employed; therefore (7.3) is computed multiple times. The main motivation of using the convolutional operation is that is equi-variant: if the input image is translated, the activations of the network in each layer will translate in the same way. More, formally an operator f is equivariant to g if:

$$f(g(x)) = g(f(x))$$

Remarkably, operators that are equivariant to shift, can be represented using the Fouries basis. Since the convolution operation can be applied in any dimension, it is common, in the presence of 2D data (images) to preserve the original input structure. The different outputs obtained using different kernels are then stacked and treated as *channels* (more details available in (Goodfellow, Bengio, and Courville, 2016b)). (ii) Pooling layers perform a subsampling of the previous layer's output, usually by averaging (*average pooling*) or taking the maximum value (*max-pooling*) over a contiguous region of values. In Fig. 7.3 a graphical explanation of max-pooling is provided.

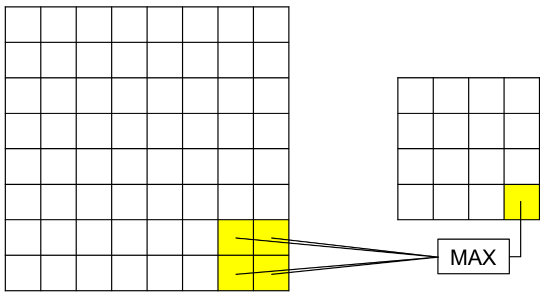


FIGURE 7.3: Representation of the max pooling operation over a two-dimensional input using 2×2 regions.

(iii) Fully Connected (FC) layers are the same employed in FNNs therefore the same description applies here. Usually FC layers are placed at the end of the network. Since the structure of a FC layer is one-dimensional, usually multi-dimensional data are *flattened* into a 1D vector for processing by this layer.

ANNs provide an approximation function $y = f^*(x;\theta)$, parametrized by a set of parameters $\theta \in \Theta$ (matrices and biases in FNNs, kernel/matrices and biases for CNNs) to an arbitrary complex continuous function f (Goodfellow, Bengio, and Courville, 2016b). The creation of the predictive model thus requires the estimation of the parameters θ that best approximate the desired output; this is achieved by minimization of a cost function defined according to the output layer properties; common choices are Mean Squared Error for regression and cross-entropy for classification. Usually, a stochastic gradient-descent (SGD) based algorithm is used, exploiting *backpropagation* (Rumelhart, Hinton, and Williams, 1988). The term stochastic derives form the fact that gradients are computed over random mini-batches of data. With the term Deep Neural Networks (DNNs) we refer to NNs with more than 2 layers (usually > 10). The success of CNNs for image recognition tasks led to the development of many variants of convolutional architectures. The most notable are the so-called Residual

Networks (ResNet) (He et al., 2016) whose internal layer has a residual form:

$$x_{t+1} = x_t + g_\theta(x)$$

whereas the last (classification) layer is a simple affine transformation $Ax + b$. Thus the global form of ResNet is:

$$f_\theta(x) = Ar_\theta(x) + b$$

where $r_\theta(x)$ can be seen as a feature extractor. We will refer to $r_\theta(x)$ as features or representations. The main advantage of the residual structure is the ability to train very deep networks with more than 1000 layers due to its implicit regularization of the landspace (Li et al., 2018). In fact, its form has natural interpretations from dynamical systems point-view (Lu et al., 2017) and optimal transport (Sonoda and Murata, 2019). We will also give an optimal transport interpretation of the role of ResNets in the context of generative frameworks and robust training. Throughout this work, we will use only ResNets and their variation called Wide-ResNets (Zagoruyko and Komodakis, 2016).

The last important ingredient of state-of-the-art architectures is Batch Normalization (BN) (Ioffe and Szegedy, 2015). Its original motivation was to alleviate the the so-called internal covariate shift of features statistics during training: the distribution of each layer's inputs changes during training, as the parameters of the previous layers change. And for the authors, this may slow down the training by requiring lower learning rates and careful parameter initialization. In order to solve this issue they define a post-activation (applied element-wise) layer:

$$y_t = \gamma \hat{x}_t + \beta$$

where, γ and β are parameters to be learnt and $\hat{x}_t$ is the whitened statistic of x_t:

$$\hat{x}_t = \frac{x_t - \mathbb{E}\, x_t}{\sqrt{\mathrm{Var}(x_t)}}$$

In practice, expectation and variance are computed over mini-batches and are update on an online fashion. BN allows for higher learning rate and faster training convergence. However, despite its pervasiveness, the exact reasons for BatchNorm's effectiveness are still poorly understood and recent empirical results show it makes the optimization landscape significantly smoother (Santurkar et al., 2018).

Throughout this thesis, we represent a dataset of N samples and K classes as $\mathcal{D} = \{x, t\}_{i=1}^N$, where $x \in \mathcal{X} \subset \mathbb{R}^p$ and $t \in \mathcal{T}$, $|\mathcal{T}| = K$ are the input and targets respectively. The classifier c is of the form $c_\theta(x) = \arg\max_{t \in \mathcal{T}} f_\theta(t|x)$ where $f_\theta(\cdot|x)$ is a parametric model with parameters $\theta \in \Theta$. The standard risk is: $\mathcal{L}_{ST}(\theta) = \mathbb{E}_{x,t \in \mathcal{D}}\, \ell\left(f_\theta(x), t\right)$ where ℓ is the standard cross-entropy loss (Negative Log Likelihood) corresponding to the class $t(x)$:

$$\ell\left(f_\theta(x), k(x)\right) = -\log p_\theta(x)_{t(x)} \tag{7.4}$$

where:

$$p_\theta(x)_k = \frac{\exp f_{\theta,k}(x)}{\sum_{j=1}^K \exp f_{\theta,j}(x)}$$

where by $(\cdot)_k$ we indicate the k-th component of the vector. In order to ease the notation, from now on will use $f_\theta(x)$ to represent both $f_\theta(x)$ itself and $p_\theta(x)$. The actual meaning will be either not necessary or clear from the context.

7.2 Optimal transport

Since the following two chapters are based on Optimal Transport (OT) theory, we will review here its main theoretical results obtained since its introduction in 1781 by Monge (Monge, 1781). For an exhaustive review of the main topics of Optimal Transportation, we suggest to the reader (Villani, 2008).

Informally, the Optimal transport theory tries to give an answer to the following question: Given a prescribed cost $c(x,y)$, how can we transport a mass from a source x to a target destination y in the optimal way? For military reasons, Monge formulated the optimal transport problem:

$$\min_T \int_{\mathcal{X}} c(x, T(x)) d\mu(x)$$

over the set of all measurable maps T such that $T_{\#}\mu = \nu$. Thus, it aims at finding the best plan to move mass μ to mass ν, fixed a cost $c(x,y) : \mathcal{X} \subset \mathbb{R}^p \times \mathcal{Y} \subset \mathbb{R}^p \to \mathbb{R}$. However, this problem is very intractable as it requires a 1-to-1 *transport map* T. In the 1940s, Kantorovich introduced the relaxation of Monge's problem (Villani, 2008). Any strategy for sending μ onto ν can be represented by a joint measure ρ on $X \times Y$, such that

$$\rho(A \times Y) = \mu(A), \rho(X \times B) = \nu(B), \tag{7.5}$$

$\rho(A \times B)$ is called a *transportation plan*, which represents the share to be moved from A to B. We denote the projection to X and Y as π_x and π_y respectively, then $\pi_{x\#}\rho = \mu$ and $\pi_{y\#}\rho = \nu$. The total cost of the transportation plan ρ is

$$\mathcal{C}(\rho) = \int_{X \times Y} c(x,y) d\rho(x,y). \tag{7.6}$$

This problem is formalized (in its relaxed form) with the Monge-Kantorovich problem. It consists in finding the plan ρ, among all the suitable transportation plans, minimizing $\mathcal{C}(\pi)$ in (7.6)

$$W_c(\mu,\nu) := \min_\rho \left\{ \mathcal{C}(\pi) = \int_{\mathcal{X}\times\mathcal{Y}} c(x,y) d\rho(x,y) \ : \ \pi_{x\#}\rho = \mu, \pi_{y\#}\rho = \nu \right\} \tag{7.7}$$

where π denote the projection operator $\pi_x : (x,y) \to x$. Its applications goes from Economics (Figalli, Kim, and McCann, 2011) to Physics (McCann, 1997). In the last decade, OT gained attention from the Machine Learning and Computer Vision fields resulting in interesting applications such as color-transfer, shape matching, image generation and many others (Peyré and Cuturi, 2018).

Problem (7.7) is called the primal formulation of the OT problem. Its dual form (Villani, 2008) is:

$$W_c(\mu,\nu) := \max_{\varphi,\psi} \left\{ \int_{\mathcal{X}} \varphi(x) d\mu(x) + \int_{\mathcal{Y}} \psi(y) d\nu(y) \right\}, \ \varphi(x) + \psi(y) \leqslant c(x,y) \tag{7.8}$$

where $\varphi : \mathcal{X} \to \mathbb{R}$ and $\psi : \mathcal{Y} \to \mathbb{R}$. Re-defining $\varphi \leftarrow -\varphi$ and $\psi \leftarrow -\psi$, (7.8) can be rewritten as:

$$W_c(\mu, \nu) := \min_{\varphi,\psi} \left\{ \int_{\mathcal{X}} \varphi(x) d\mu(x) + \int_{\mathcal{Y}} \psi(y) d\nu(y) \right\}, \ \varphi(x) + \psi(y) \geqslant -c(x,y) \tag{7.9}$$

In this (more counter-intuitive) formulation $-c(x,y)$ is a negative cost, that is, a utility or profit $b(x)$: the constraint $\varphi(x) + \psi(y) \geqslant b(x,y)$ requires that the total profit is at least $b(x,y)$. A simple way to ensure the optimality constraint is satisfied is to set a $\psi(y)$ given by the so-called c-transform:

Definition 2 (c-transform). Given a real function $\varphi : \mathcal{X} \to \mathbb{R}$, with $\mathcal{X}$ compact and c strictly convex, the c-transform of φ is defined by

$$\varphi^c(y) = \max_{x \in X} -c(x,y) - \varphi(x) = \min_{x \in \mathcal{X}} c(x,y) + \varphi(x) \tag{7.10}$$

.

The constrained problem (7.9) can written as the unconstrained problem:

$$W_c(\mu, \nu) := \min_{\varphi} \left\{ \int_{\mathcal{X}} \varphi(x) d\mu(x) + \int_{\mathcal{Y}} \varphi^c(y) d\nu(y) \right\} \tag{7.11}$$

Let $\varphi_{OT}(x)$ the potential solving (7.11), then the c-transform corresponds to the *transportation map* (Villani, 2008) $T(x)$ from μ to ν:

$$T(x) = \arg\max_{y \in \mathcal{Y}} -c(x,y) - \varphi_{OT}(y) \tag{7.12}$$

and

$$T(y) = \arg\max_{x \in \mathcal{X}} -c(x,y) - \varphi^c_{OT}(x)$$

is the "backward" optimal map from ν to μ. In case of quadratic cost $c(x,y) = \frac{1}{2}\|x-y\|^2$, if we agglomerate $\frac{1}{2}\|x\|^2$ and $\frac{1}{2}\|y\|^2$ with their respective potentials $\varphi(x)$ and $\psi(y)$, the c-transform specializes to the Fenchel-Lengendre transform:

$$\varphi^*(y) = \max_{x \in X} x^\top y - \varphi(x) \tag{7.13}$$

that is convex in y (Boyd and Vandenberghe, 2004). Finally, we introduce the so-called Wasserstein metrics, defined in the following:

Definition 3. Let $\mathcal{X}$ a Polish space endowed with a metric d. For $p \geqslant 1$, let $P_p(\mathcal{X})$ denote the collection of probability measure with finite p-th moment. We define the p-Wasserstein distance between μ and ν as:

$$W_p(\mu, \nu) := \left(\inf_{\rho} \int_{\mathcal{X} \times \mathcal{X}} d(x,y)^p d\rho(x,y)) \right)^{\frac{1}{p}} \tag{7.14}$$

such that $\pi_{x\#}\rho = \mu, \pi_{y\#}\rho = \nu$.

When $p = 1$, we have the equivalent Kantorovich-Rubinstein formulation:

$$W_1(\mu, \nu) := \min_{\varphi \in Lip(\varphi) \leqslant 1} \left\{ \int_{\mathcal{X}} \varphi(x) d\mu(x) - \int_{\mathcal{Y}} \varphi(y) d\nu(y) \right\} \tag{7.15}$$

where the maximum is taken over the set of 1-Lipshitz functions. Informally, if c is a distance, a function φ is c-convex function if and only if is 1-Lipshitz. Thus the c-transform is $\varphi^c = -\varphi$ and the transport map becomes:

$$T(x) = \arg\max_{y \in \mathcal{Y}} -c(x,y) + \varphi_{OT}(y) \tag{7.16}$$

7.2.1 Brenier's theorem

Brenier (Brenier, 1991) gave an important contribution to the Monge-Kantorovich problem, providing an explitit form of the transportation map:

Theorem 4 (Brenier's theorem). *Let μ and ν on a compact domain $\mathcal{X} \subset \mathbb{R}^p$. Assume μ does not five mass to small sets, and that the cost $c(x,y) = h(x-y)$ with h strictly convex. Then, there exists a unique transport map with form:*

$$T(x) = x - (\nabla h)^{-1}(\nabla \varphi(x)).$$

When $c(x,y) = \frac{1}{2}|x-y|^2$, we have

$$T(x) = x - \nabla\varphi(x) = \nabla\left(\frac{x^2}{2} - \varphi(x)\right) = \nabla u(x).$$

In this case, the Brenier's potential u and the Kantorovich's potential φ is related by

$$u(x) = \frac{x^2}{2} - \varphi(x). \tag{7.17}$$

This theorem says that once we have our estimate of the convex potential φ, then the transportation is a simple function of its gradient evaluated in the source point x.

7.2.2 Displacement interpolation

A fundamental concept known in optimal transport theory is *displacement interpolation* of probability measures.

Definition 5 (Villani (2008)). Let μ and ν two probability measures on $\mathbb{R}^p$. Consider the cost $c(x,y) = \|x-y\|^2$. Let $\nabla\varphi$ be the optimal transport map such that $\nabla\varphi_{\#}\mu = \nu$. Define:

$$\rho_\tau = [(1-\tau)Id + \tau\nabla\varphi]_{\#}\,\mu$$

The path defined by probability measures $(\rho_\tau)_{0\leqslant\tau\leqslant 1}$ is the minimum cost path (geodesic) from μ to ν. Moreover, we have:

$$\rho_\tau = \arg\min_{\varrho}(1-\tau)\mathcal{W}(\mu,\varrho) + \tau\mathcal{W}(\varrho,\nu)$$

Displacement interpolation and displacement convexity was fundamental concept to show that certain potential functional used in physics, such as for interacting gas, that are not convex in R^p, they are convex in the the space of Wasserstein space of interpolations (McCann, 1997).

7.2.3 Discrete Optimal Transport

In many real-world applications, we do not have (access to) continuous probability measures, but usually to empirical estimates or directly discrete measures. Suppose the existence of two discrete spaces $\mathcal{X}$ and $\mathcal{Y}$ with measures:

$$\mu = \sum_{i=1}^{n} \delta_{x_i}, \qquad \mu = \sum_{j=1}^{n} \delta_{y_j}$$

where δ represents the Dirac delta. Any measure on the set of admissible plans must be represented by a bi-stochastic matrix $n \times n$ $\pi = (\pi_{i,j})$, which means that:

$$\sum_i \pi_{i,j} = 1 \qquad \text{and} \qquad \sum_j \pi_{i,j} = 1$$

The transportation cost is represented by a $n \times n$ matrix C:

Definition 6 (Cost matrix $C^{\odot p}$). A symmetric positive semi-definite matrix $C \in \mathbb{R}_+^{n \times n}$ defines a pseudo-Riemannian metric on the domain, an entry $C_{i,j}$ is the cost of transporting unit probability mass from class i to class j. Note that $C_{i,i} = 0$. The notation $C^{\odot p}$ denotes the element-wise p^{th}-power of C.

Thus, in the discrete domain, p-Wasserstein distance between μ and ν for $p \in [1, \infty)$ becomes:

$$\mathrm{W}_p^p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \left\langle \pi,\ C^{\odot p} \right\rangle \tag{7.18}$$

where $\Pi(\mu, \nu) = \left\{ \pi \in \mathbb{R}_+^{K \times K} :\ \mu = \pi \mathbb{1},\ \nu = \pi^\top \mathbb{1} \right\}$ is the set of joint probability distributions with q as the right marginal and q' as the left marginal; $\mathbb{1}$ denotes the all-one vector and $\langle \cdot, \cdot \rangle$ is the Frobenius inner product on matrices. For $0 < p \leqslant 1$, the Wasserstein distance in (7.18) is defined to be $\mathrm{W}_p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \langle \pi,\ C^{\odot p} \rangle$; note the absence of p^{th} power on the left-hand side. For any separable complete metric space $(\mathcal{X}, d)$ and $p > 0$, the metric space $(\mathcal{P}_p, \mathrm{W}_p)$ is complete and separable, $\mathcal{P}_p$ being the set of probability distributions supported on $\mathcal{X}$ (Ambrosio, Gigli, and Savaré, 2008).

Problem (7.18) takes $\mathcal{O}(K^3)$ operations to be solved using linear programming or interior point methods. (Cuturi, 2013) proposed a smoothed alternative to (7.18) by adding a convex negative entropic term

$$^{\lambda}\mathrm{W}_p^p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \left\langle \pi, C^{\odot p} \right\rangle - \lambda^{-1} \mathrm{H}(\pi), \tag{7.19}$$

$\mathrm{H}(\pi) = -\sum_{i,j=1}^{n} \pi_{i,j} \log \pi_{i,j}$ that enables an efficient algorithm based on Sinhorn-Knopp iteration (Sinkhorn, 1964) to approximate π^*. Large values of λ give better approximation to the exact distance W_p^p and it can be shown that $^{\lambda}\mathrm{W}_p^p$ converges to W_p^p as $\lambda \to \infty$ (Peyré and Cuturi, 2018).

7.2.4 Gradient flows

Let $F : \mathbb{R}^p \to \mathbb{R}$ be a function and an initial point $x_0 \in \mathbb{R}^p$. A gradient flow (Santambrogio, 2015) is an evolution starting from x_0 and always moving in the direction where F decreases (increases) the most, thus "gradually minimizing (maximizing)" φ.

More precisely, it is just the solution of the Cauchy problem:

$$\begin{cases} \dot{x}(t) = & -\nabla F(x(t)) \\ x(0) = & x_0 \end{cases}$$

if it is a minimizing flow or

$$\begin{cases} \dot{x}(t) = & \nabla F(x(t)) \\ x(0) = & x_0 \end{cases}$$

if it is a maximizing flow. In the following we will use the maximizing gradient flow. The minimizing one is obtained by trivial changes.

The discritization of the flow is given by the sequence:

$$x_{k+1}^{\tau} \in \arg\max_x F(x) - \frac{|x - x_k^{\tau}|^2}{2\tau}$$

This discretized formulation can easily be adapted to metric spaces. If we have the metric space $(\mathcal{X}, d)$ and a upper semi-continuous function (u.s.c.) $F : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ bounded from above, we can define:

$$x_{k+1}^{\tau} \in \arg\max_x F(x) - \frac{d(x, x_k^{\tau})^2}{2\tau} \tag{7.20}$$

This argument can be extended also to functionals $F : P(\Omega) \to \mathbb{R}$, with Ω compact, in the 2-Wasserstein space $\mathcal{W}_2$.

$$x_{k+1}^{\tau} \in \arg\max_x F(x) - \frac{W_2^2(x, x_k^{\tau})}{2\tau} \tag{7.21}$$

7.3 f-divergences

Another tool we will consider are the so-called f-divergences. Given two distributions μ and ν defined on the domain $\mathcal{X}$, we define the *f-divergence*,

$$D_f(\mu \| \nu) = \int_{\mathcal{X}} f\left(\frac{d\mu}{d\nu}\right) \mathrm{d}\nu, \tag{7.22}$$

where the *generator function* $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex, lower-semicontinuous function satisfying $f(1) = 0$. The KL-divergence is a particular case of (7.22):

Definition 7. Let $\mu, \nu \in P(\mathbb{R}^p)$ The KL-divergence is given by:

$$D_{KL}(\mu, \nu) = \int \log \frac{d\mu}{d\nu} d\mu$$

if μ is absolutely continuous with respect to ν ($\mu \ll \nu$) and $+\infty$ otherwise.

Thus the D_{KL} is obtained with the generation function $f(t) = t \log t$. For a summary of the most common f-divergences please see (Nowozin, Cseke, and Tomioka, 2016).

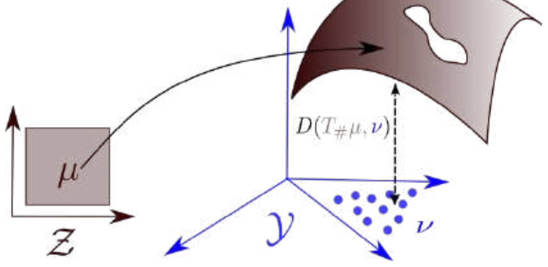


FIGURE 7.4: Illustration of generative framework using the Minimum Kantorovitch Estimator or f-divergences.

f-divergences can be expressed by their dual formulation:

$$D_f(\mu\|\nu) = \sup_{g\in\mathcal{G}} \left\{ \int g d\mu - \int f^*(g) d\nu \right\}, \tag{7.23}$$

where $\mathcal{G}$ is an arbitrary class of functions $g : \mathcal{X} \to \mathbb{R}$ and f^* is Legendre-Fenchel transform. In view of this, let us remind the dual variational formulation of the KL divergence:

Proposition 8. *Let $\nu, \mu \in P(\mathbb{R}^p)$. Then:*

$$D_{KL}(\mu\|\nu) = 1 + \sup_{g\in\mathcal{G}} \left\{ \int_{R^p} g d\mu - \int_{R^p} e^g d\nu \right\} \tag{7.24}$$

Proof. This proposition is result of the convexity of $-\log x$ and by recognizing that it can be obtained by the Legendre transform:

$$-\log x = \sup_{y<0}\{xy + 1 - \log(-\frac{1}{y})\} = \sup_g = \{g - xe^g\} + 1$$

■

7.4 Generative Adversarial Networks

The objective of generative frameworks is to learn a probability distribution by a parametric model from which it is easy to sample. In the context of DL, the two most well-established generative frameworks are Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) and Variational Auto-Encoders (Kingma and Welling, 2013). Here we will overview only GANs, since they are mainly related to this thesis which is focused on discriminative models.

The idea of common generative frameworks is quite simple: choose (for GANs) or learn (for VAEs) a distribution μ from which it is to sample and map it to a target distribution ν through a parametric model.

More precisely, one wants to find the parametric map T_ξ that transports μ to ν solving the problem:

$$\min_{T_\xi} D(T_{\#}\mu\|\nu) \tag{7.25}$$

where $D = D_f$ if using f-divergences or $D = W_c$ if using Wasserstein metrics. The latter are called Minimum Kantorovich Estimators (MKE). In general, the maximum likelihood estimator is undefined or difficult to compute (because the support of the measures $T_{\#}\mu$ are singular) while MKEs are preferable because they are always well defined (Arjovsky, Chintala, and Bottou, 2017). Moreover, when the target density ν is not known and it is in a high-dimensional setting, the divergence cannot be easily estimated.

GANs try to overcome these issues by playing an adversarial game. In fact, a GAN model consists of a generator T and a discriminator φ, both represented as deep networks. The generator samples from a simple distribution and generates samples, while the discriminator estimates the probability that a sample came from the training data rather than T. Both generator and the discriminator are trained simultaneously. For a given f-divergence the f-GAN problem amounts to solve (Nowozin, Cseke, and Tomioka, 2016):

$$L[\theta, \xi] = \min_{\theta} \max_{\xi} \left\{ \int \varphi_\theta(x) d\nu(x) - \int \varphi^*_\theta(T_\xi(x)) d\mu(x) \right\} \tag{7.26}$$

where f^* represents the Legendre transform defined as shown in Sec. 7.3 and μ is usually an isotropic noise, such as a Gaussian in $\mathbb{R}^d$ and $d \ll p$. Moreover, the form φ and its corresponding domain depends on the D_f used. For example, the original GANs, that are equivalent to Jensen-Shennon divergence (up to a translation) assume the form:

$$\min_{\theta} \max_{g} L[\varphi, \xi] \equiv - \mathop{\mathbb{E}}_{Z \sim \mu} \left[\log \left(1 - f_\theta(T_\xi(Z)) \right) \right] - \mathop{\mathbb{E}}_{X \sim \nu} \log f_\theta(X) \tag{7.27}$$

and f_θ is given by the sigmoidal output of the distriminator. With an optimal transport point of view, this consists in a minimax where:

- fixed the generator T, the player φ will (try to) to balance the mass: it will move mass from the regions where T is large (that is, $T_{\#}\nu(y) \geqslant \nu(y)$) to regions where T is small (that is, $T_{\#}\nu(y) \leqslant \nu(y)$);
- conversely fixed φ, T will try to reach the optimal $T^* = \log f'\left(\frac{T_{\#}\mu(y)}{\nu(y)}\right)$, where we remember that in this context f represents the generator function of the divergence and not the output f_θ of the network. The previous term is large where there is a lack of mass and it is small where there an extra portion of mass.

The game converges where the two players do not have mass to move anymore. In other words, the game finishes when the generator is not able to fool anymore the discriminator and the latter is not able to distinguish a real sample (e.g. image) from a one generated by T. It is easy to see that the point of equilibrium is a Nash equilibrium.

GANs are able to generate very realist images (see for example Fig. 7.5). Furthermore, we would like to remark that they do not need an explicit expression of the distribution of real data. Due to its effectiveness, this framework received much attention in numerous computer vision tasks such as image inplainting (Pathak et al., 2016; Yeh et al., 2017), object detection (Radford, Metz, and Chintala, 2015), semantic segmentation (Luc et al., 2016), image super resolution (Ledig et al., 2017). These are only very few examples of fields that have benefited from using GANs.

The major limitations of GANs are mode collapse and the instability of training. Mode collapse simply means that the training may converge to regions where data is not defined. This usually happens when trying to fit multi-modal distributions. One

FIGURE 7.5: GANs can generate realist images. Images are adapted by Brock, Donahue, and Simonyan (2018).

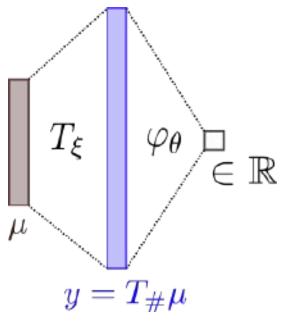


FIGURE 7.6: Illustration of GANs model.

way to partially fix this issue is to make, at every step of generator update, multiple steps of gradients update for the discriminator in order to assure optimality of φ. In this way, gradient updates for the generator are more reliable. Instead, instability also depends on the divergence used: as we mentioned above for maximum likelihood approaches, common divergences such KL or Total Variation, are not always well defined. In fact, when are dealing with distributions supported by low dimensional manifolds (like images), the KL distance is not defined (or simply infinite) because the model manifold and the true distribution's support may have negligible intersection. A remedy was was to inject noise to input. However, a common noise was a Gaussian noise that tends to make the image blurry. In order to tackle this issues, (Arjovsky, Chintala, and Bottou, 2017) introduced Wasserstein GANs (WGANs) which are simply GANs where the employed divergence is the 1-Wasserstein distance (7.15), which is always well-defined.

Chapter 8

Robustness of DL models

In this chapter will discuss the main problem that motivated the research presented in Chapter 9: the robustness to input perturbations. Moreover, we will present the most effective framework to achieve robust models as it will be the baseline for our experiments and new algorithms.

8.1 What is an adversarial example?

Szegedy et al., 2013 was the first paper which discovered that DNNs are susceptible to adversarial perturbations, i.e., modifications to the input image that although imperceptible to the human eye cause the network to mis-classify, confidently, the image. In Fig. 8.1 we provide an example of adversarial perturbations adapted from Moosavi-Dezfooli, Fawzi, and Frossard, 2016.

At the time, this worst-case perturbation $x^\star$ around a point x was computed by a first-order procedure called Fast Gradient Sign Method (FGSM). FGSM is an attack for an $\|\cdot\|_\infty$-bounded adversary and computes an adversarial example as

$$x^\star = x + \underbrace{\epsilon \operatorname{sign} \nabla_x \ell\left(f_\theta(x), t\right)}_{\delta} \tag{8.1}$$

where the perturbation has to respect the condition $\|\delta\|_\infty \leqslant \epsilon$. Of course, this attack ca be generalized to other norms. We can interpret this attack as a simple one-step scheme for maximizing the inner part of the saddle point formulation. After the original work, many other attacks were proposed, e.g. DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard, 2016) which consists on a iterative first-order projections on the boundaries. Despite its simplicity, it is very effective to find bounded perturbations. Adversarial perturbations are easy to synthesize and they may even generalize across different networks (Moosavi-Dezfooli et al., 2017). This suggests surprising vulnerabilities in these state-of-the-art classifiers and it has resulted in a flurry of activity towards understanding this phenomenon (Schmidt et al., 2018; Fawzi, Fawzi, and Frossard, 2018), building robustness and defenses against it (Goodfellow, Shlens, and Szegedy, 2014; Madry et al., 2017), as also discovering new attacks (Carlini and Wagner, 2017; Athalye, Carlini, and Wagner, 2018; Papernot, McDaniel, and Goodfellow, 2016; Papernot et al., 2016).

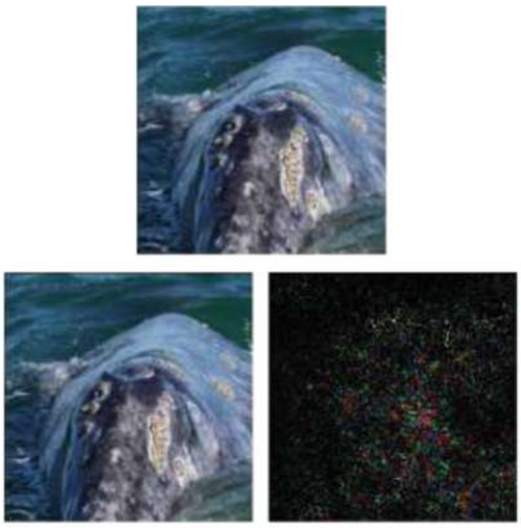


FIGURE 8.1: Adversarial example. Top row: an original image x classified as a *whale*. Bottom row: (left) adversarial image $x + \delta$ classified as a *turtle*, (right) δ perturbation found by DeepFool.

8.2 Adversarial training

All of these approaches used to find adversarial perturbation can be considered as an attempt to solve the maximization:

$$\mathop{\mathbb{E}}_{x,t \in \mathcal{D}} \max_{\delta \in \mathcal{M}} \ell\left(f_\theta(x+\delta), t\right) \tag{8.2}$$

where $\mathcal{M}$ a opportune set, that commonly consists on the ball

$$\mathcal{M} = \mathcal{M}_\epsilon = \left\{\delta : \|\delta\|_p \leqslant \epsilon\right\}$$

.
In order to solve the inner problem of (8.4), a powerful adversary is a multi-step variant, which is essentially projected gradient descent (PGD) on the negative loss function:

$$x^{t+1} = \Pi_{x+\mathcal{M}_\infty}\left(x^t + \nabla_x \ell\left(f_\theta(x), t\right)\right) \tag{8.3}$$

where $\Pi_{x+\mathcal{M}_\infty}$ is the projection operator that guarantees that adversary is bounded on the the fixed ball.
Thus, with a robust optimization point of view, the most natural way to obtain a robust model is to solve the saddle point problem (Madry et al., 2017):

$$\mathcal{L}_{rob}(x;\theta) = \min_\theta \mathop{\mathbb{E}}_{x,t \in \mathcal{D}} \max_{\delta \in \mathcal{M}} \ell\left(f_\theta(x+\delta), t\right) \tag{8.4}$$

Thus the previous problem corresponds to replacing the input point x by its corresponding adversarial perturbations $x^\star$ and then training the network on the new perturbed input. However, the gradient $\nabla_\theta \ell\left(f_\theta(x^\star), t\right)$ is not guaranteed to be a valid (descent) direction for the saddle point problem (8.4). In general, this guarantee is provided by the Danskin's theorem (see Appendix A.1) which requires the loss ℓ to be continuously differentiable in θ. However, due to ReLU and max-pooling, the loss function is not continuously differentiable. Nevertheless, one argument also provided by Madry et al., 2017 is that since the set of discontinuities has measure zero, we can assume that this will not be an issue in practice, as we will never encounter the problematic points. Moreover, Danskin's theorem assumes that the inner maximization is solved which cannot be guaranteed (due to not concavity of the inner problem) and PGD will converge to local maxima. However, if the inner maximum is a true adversarial example for the network, then SGD using the gradient at that point will decrease the loss value at this particular adversarial examples, thus making progress towards a robust model and making the network progressively smoother. In fact, as empirical findings show, local maxima found from PGD are already good in the sense the they maximize the loss very well.

Solving the saddle-point problem allows to achieve adversarial robustness. However, the the no free lunch theorem holds: robustness has a price, that is, robust models have low classification accuracy even on clean data. This is a fundamental trade-off under certain analyses (Tsipras et al., 2018b; Fawzi, Fawzi, and Fawzi, 2018). For example, on CIFAR-10, with the $\|\cdot\|_\infty$ norm, an $\epsilon = 8/255$ causes a decrease of accuracy of 8%, passing from 96% to 88% for a W-28-10. This is the key point that motivated the next chapter: an adversarially robust classifier with low accuracy is unlikely to be used in practice, since practical applications require both.

Chapter 9

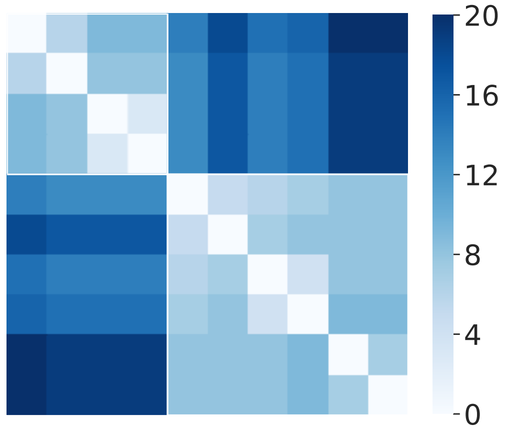
Directional robustness

In many real-world applications of Machine Learning it is of paramount importance not only to provide accurate predictions, but also to ensure certain levels of robustness. Adversarial Training is a training procedure aiming at providing models that are robust to worst-case perturbations around predefined points. Unfortunately, one of the main issues in adversarial training is that robustness w.r.t. gradient-based attackers is always achieved at the cost of prediction accuracy. In this chapter, a new algorithm, called Wasserstein Projected Gradient Descent (WPGD), for adversarial training is proposed. WPGD provides a simple way to obtain cost-sensitive robustness, resulting in a finer control of the robustness-accuracy trade-off. Moreover, WPGD solves an optimal transport problem on the output space of the network and it can efficiently discover directions where robustness is required, allowing to control the directional trade-off between accuracy and robustness. The proposed WPGD is validated in this work on image recognition tasks with different benchmark datasets and architectures. Moreover, real world-like datasets are often unbalanced: this chapter shows that when dealing with such type of datasets, the performance of adversarial training are mainly affected in term of standard accuracy.

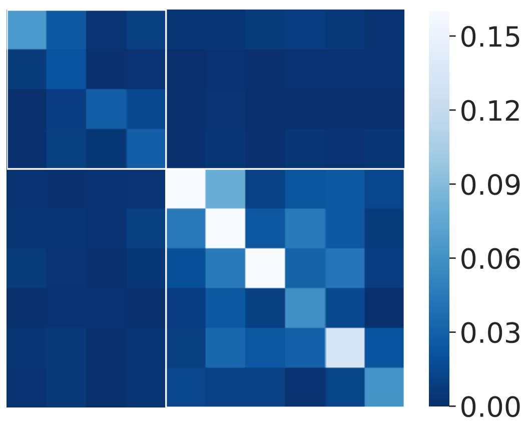
9.1 Related work

As we summarized in Sec. 8.2, the most common approach to provide models robust to adversarial examples is *Adversarial Training* (Madry et al., 2017), a training procedure belonging to the class of minimax problems, in which a inner loop finds the worst-case data point $x^{\star}$ and the outer loop minimizes the target loss on $x^{\star}$. An adversarially robust classifier with low accuracy is unlikely to be used in practice, practical applications require both. Although much effort has been devoted to theoretically understand robustness, its practical consequences in industrial applications received few attention from the literature only recently (Ibitoye, Shafiq, and Matrawy, 2019). In particular, in many industrial applications such as IoT, autonomous driving (Qayyum et al., 2019) and predictive maintenance (Susto et al., 2014), errors and faults have different priorities and importance: for example, if a recognition system of an autonomous car misclassifies a cat as a dog there should be reasonably no damage. On the contrary, if it classifies a human as a truck or a dog, the decision may cause dramatic consequences.

This chapter is mainly related to Madry et al., 2017; Tsipras et al., 2018a. Although they give theoretical and practical results on the connection between robustness and accuracy for adversarial training, they don't analyze how the accuracy gap is distributed. They also argue that adversarial training requires extra capacity in order to build complex boundaries (Kolter and Wong, 2017). In contrast, Moosavi-Dezfooli et al., 2018 have recently argued that adversarial training leads to flatter decision boundaries and in fact, explicitly penalizing the curvature of the decision boundary is a good technique to train robust classifiers. Our results corroborate these findings. The accuracy gap of adversarially trained networks with respect to standard cross-entropy trained networks can be explained, very well as our experiments show, by the network getting these pairs of classes incorrect. Semantic metrics, e.g., those derived from WordNet (Miller, 1995) to aid visual classification have been popular to introduce a new data-modality in standard supervised learning (Deng et al., 2010; Deng et al., 2014). We identify the inherent visual metric that the network induces while being trained using cross-entropy loss or the adversarial loss. Lastly, our method of using an optimal transport formulation to impose a metric on the label space of deep networks bears close resemblance to the work of Frogner et al., 2015. This work uses the Wasserstein loss computed using the Sinhorn-Knopp iteration to predict multi-label images. The present work is the first to use the optimal transport formulation to induce a cost-sensitive adversarial training of deep networks. Further, for single-label images, we show that the optimal transport problem has a closed form solution which makes it computationally equivalent to the cross-entropy loss; this simple but powerful property may be of independent interest for problems like hierarchical classification (Girshick et al., 2013; Wu, Tygert, and LeCun, 2017; Bagherinezhad et al., 2018).



(A)



(B)

FIGURE 9.1: CIFAR-10. Fig. 9.2a is a matrix of pairwise distances between classes as computed by averaging a first-order estimate of the distance of the images to the decision boundaries. Classes from top to bottom are: airplane, car, bird, cat, deer, dog, frog, horse, ship, truck. Fig. 9.2b shows the difference in the confusion matrix of a Wide Residual Network trained using Projected Gradient Descent (PGD) and that of the standard cross-entropy loss. The accuracy gap is well-explained by such a "visual metric"; the correlation of entires in the two matrices is -0.65.

In view of these issues, the main contributions of this chapter are:

- we show that the quantitative and qualitative difference between robust and standard models correlates with the visual metric of classes, ie. it is aligned with the human notion of distance between classes. Adversarially trained networks learn to (mostly) ignore fine-grained classification and confuse classes with samples that are close to the decision boundaries. This result is corroborated by Moosavi-Dezfooli et al., 2018 where it is shown that adversarial training leads to boundaries with low curvature;
- we show that robust models are less confident in their predictions than standard models are;

- inspired by the previous observation, we present Wasserstein Projected Gradient Descent (WPGD), an algorithm for adversarial training of deep networks. WPGD improves the efficiency of the inner loop in gradient-based defences such as Projected Gradient Descent (PGD). WPGD formulates an optimal transport problem on the label space with the underlying metric given by the distances of the classification boundaries between classes. This metric guides the search for adversarial perturbations towards classes that are visually dissimilar. It is shown that training deep networks using WPGD is effective in shaping boundaries to maintain direction robustness where required will maintaining accuracy on similar classes.

Moreover, we would like to notice that, although our experiments regard image recognition tasks, the WPGD framework can be easily extended to other types of data such as time-series.

The rest of this chapter is organized as follows. We discuss building blocks of our approach, viz., estimating the distance to the boundaries and optimal transport in Sec. 9.2. Properties of adversarial training are discussed in Sec. 9.3, Sec. 9.4 discusses the WPGD algorithm and experimental results on MNIST (LeCun et al., 1998), CIFAR-10 and Tiny Imagenet datasets for different deep networks. Related work and discussion are provided in Sec. 9.1 and Sec. 9.6 respectively.

9.2 Building blocks

This section describes the building blocks of our approach. We will refer to the standard cross-entropy loss as ℓ_{CE} in order to distinguish it with the wasserstein loss Def. 13 and to $c_\theta(x) = \arg\max_{t \in \mathcal{T}} f_\theta(t|x)$ as the predicted class. We remind that adversarial training consists on solving the saddle-point problem:

$$\min_{\theta} \mathop{\mathbb{E}}_{X} \left[\max_{x' \in \mathcal{M}(x)} \ell_{\text{CE}}(x', t;\, \theta) \right]; \tag{9.1}$$

In this work, we only consider $\mathcal{M}(x) = \mathcal{M}_\infty(x) = \{x' : \|x' - x\|_\infty \leqslant \epsilon\}$ to be the infinity-norm ball around x to obtain an algorithm based on projected gradient descent (**PGD**) (Boyd and Vandenberghe, 2004). We would like to remark that the theoretical properties described in the following are generally applied to general setting and not only Euclidean perturbations. We refer to *natural error* (NE) and *adversarial error* (AE) as the errors obtained with natural images and adversarial images, respectively. In this thesis, we only use ℓ_∞ perturbations for all the experiments regarding real datasets while we use ℓ_2 perturbations for the synthetic example in Sec. 9.3.4.

9.3 Properties of adversarial training

In this Section some effects and properties of adversarial training on various aspects are reported. Such aspects are:

- the qualitative and quantitative description of classification errors, measured by the accuracy gap (Sec. 9.3.1);
- unbalanced classification problems (Sec. 9.3.2);
- the characterization of output confidence (Sec. 9.3.3);

- the characterization of boundaries (Sec. 9.3.4).

The aforementioned effects are supported by experiments reported in this Section.

The properties and effects of adversarial training reported here have motivated WPGD that will be presented in the following Section.

9.3.1 Accuracy gap

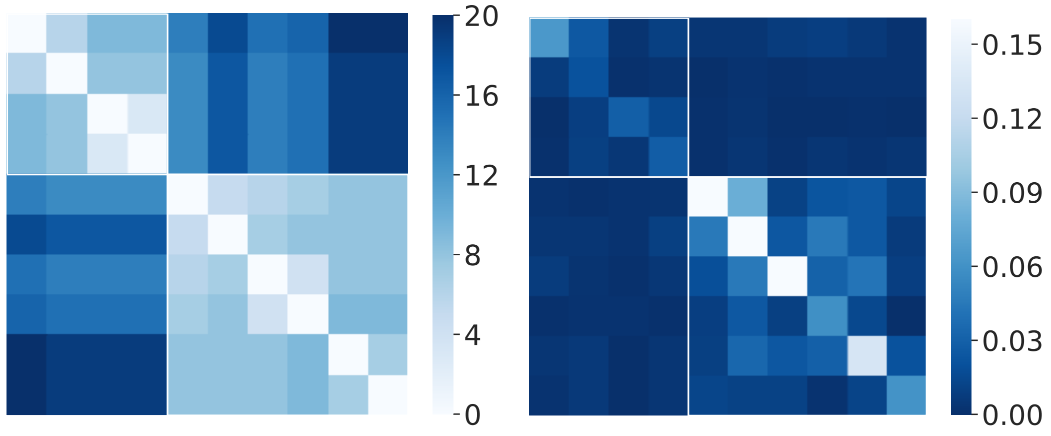
In order to ease the understanding of the results on this Section, the notion of *accuracy gap* is defined as the following:

Definition 9. Let C_{pgd} and C_{ce} be the confusion matrices of robust and standard models, respectively. The *accuracy gap* G is defined as the absolute difference between the confusion matrices:

$$G = |C_{pgd} - C_{ce}|$$

Although it is known that robustness is obtained at cost of accuracy (Madry et al., 2017; Tsipras et al., 2018b), it is not still clear in the literature whether this gap can be mitigated[1]. In this work, a first step into tackling this problem is taken by studying how errors are distributed between images and classes: it is shown in the following that mis-classification errors are distributed following the *visual metric*, meaning that robust networks tend to destroy fine-grained classification. Qualitatively, the visual metric is a distance between classes that can be easily interpreted by humans. One approach for defining such visual metric is to employ the distance from boundaries of a deep neural network: in fact, Saxe, McClelland, and Ganguli, 2019 showed that NNs learn representations that are well-aligned with our idea of visual similarity.

Due to high-dimensionality of input, obtaining a good approximation of the visual metric is not easily feasible. However, it can be replaced by the semantic metric provided by WordNet (Miller, 1995), which is a good proxy for the visual metric as also showed by Deng et al., 2010. For MNIST, we use a linear classifier on the input pixels whereby we can compute the boundaries accurately.



(A) (B)

FIGURE 9.2: CIFAR-10 dataset. In panel (a) it is reported a matrix of pairwise distances between classes. Classes are (top to bottom): airplane, car, bird, cat, deer, dog, frog, horse, ship, truck. Panel (b) shows the accuracy gap between a Wide Residual Network (Zagoruyko and Komodakis, 2016) trained using PGD and one trained with the standard cross-entropy loss.

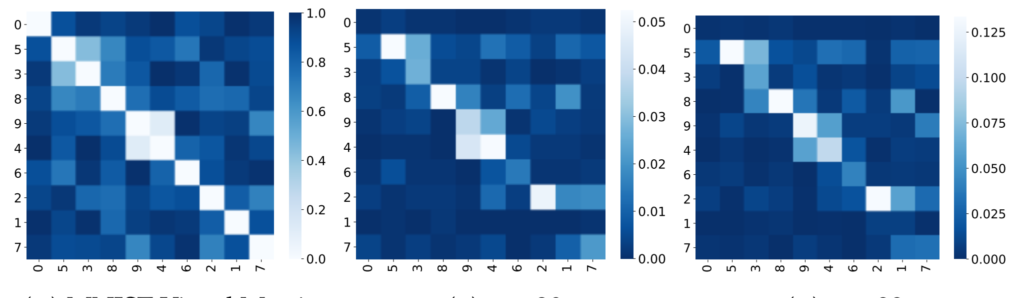
[1]On MNIST dataset, high capacity networks reduce the accuracy gap to near zero. However, in more complex datasets, such as CIFAR-10, this gap exists even with very large networks.

Fig. 9.2 illustrates results for CIFAR-10. In particular, Fig. 9.2b shows the accuracy gap between a Wide Residual Network (Zagoruyko and Komodakis, 2016) trained using PGD and one trained with the standard cross-entropy loss. From this figure, it is easy to see a visual correlation between metric and accuracy gap. Interestingly, the errors that are explained by such metrics, correspond to classes which are visually similar. For instance, Fig. 9.2 shows a gap on the pair *bird-airplane* which are visually similar but semantically different. Analogously, in Fig. 9.3, Fig. 9.4 and Fig. 9.5 it is shown the WordNet metrics and the relative accuracy gaps for MNIST, Tiny-Imagenet and CIFAR-100, respectively. Similar results are identifiable also for these datasets. In fact, regarding MNIST, not surprisingly, digits "0" and "1" hardly fool each other. The most similar digits are "4" and "9": in fact, a small manipulation of such digits can be sufficient to make them indistinguishable. Also, regarding CIFAR-100, as an example, from indices 8-11 there is an evident cluster composed by the classes *man*, *boy*, *woman* and *girl*. Other very connected classes are *bridge*, *skyscraper*, *house*, *castle* and *road*. Moreover, there are animals that are semantically different but which are visually similar, such the couple 32-90 that are *seal* and *otter* respectively. The bottom-right cluster represents flowers and plants.

In Table 9.1 a quantitative measure (supporting the aforementioned 'visual' results) of the correlation between accuracy gap and relative metric is provided. The minus sign is due to the fact that confusion matrices and distances are inversely correlated: when the values of diagonal increase of the confusion matrices, then the distance between classes decreases, on average. For MNIST the correlation is higher since we are using an approximation of the actual visual metric, while for CIFAR-10 and CIFAR-100 the correlation is lower because some pairs, for example, bird-airplane are semantically different. Moreover, it is remarked that with high output dimension, the correlation decreases even when there are well-correlated structures.

	MNIST	**CIFAR-10**	**CIFAR-100**	**Tiny-Imagenet**
Correlation	-0.88	- 0.65	-0.35	-0.22

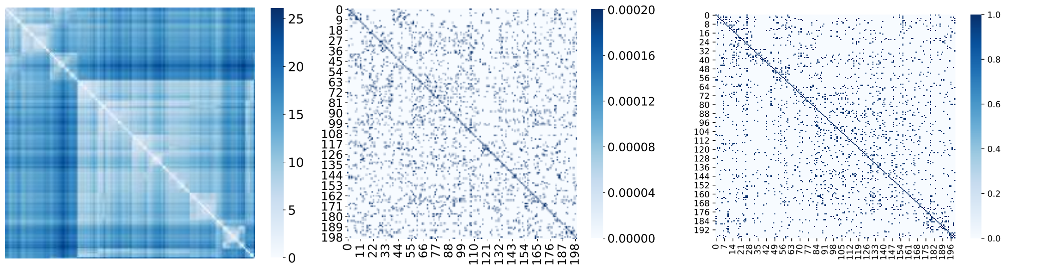
TABLE 9.1: Correlation ρ between accuracy gap and relative metric for all the datasets.



(A) MNIST Visual Metric (B) $\epsilon = 20$ (C) $\epsilon = 38$

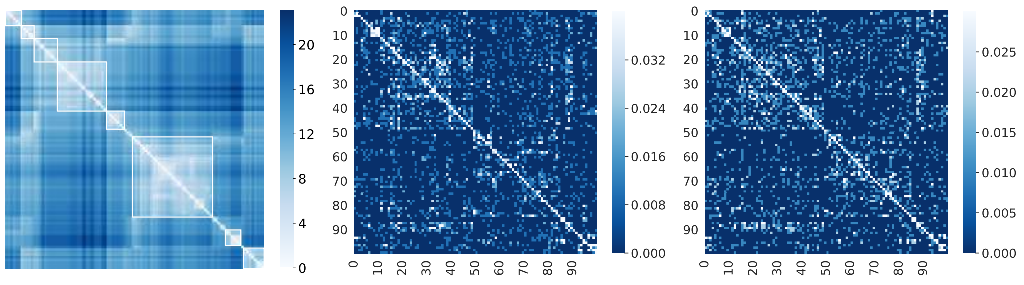
FIGURE 9.3: MNIST dataset: Accuracy gap G between baseline model and PGD-trained model with $\epsilon = 20$ Fig. 9.3b and $\epsilon = 38$ Fig. 9.3c. The gap in accuracy caused by PGD training correlates with the visual metric. This causes the network to be less effective in fine-grained classification.

Given these premises and observations, the following conjecture can be made: when the number of classes is high then boundaries among similar classes becomes



(A) WordNet TinyImagetNet (B) Accuracy gap (C) Confusion matrix of robust network.

FIGURE 9.4: Accuracy gap between baseline model and PGD-trained model with $\epsilon = 8$ for TinyImagetNet.



(A) WordNet CIFAR-100 Metric (B) All-CNN (C) W-28-10

FIGURE 9.5: Accuracy gap between baseline model and PGD-trained model with $\epsilon = 8$ for CIFAR-100. The main cluster in the upper-left part of Fig. 9.5b identifies animals in general.

more complex. Thus, as an ablation study, two 2-classes problems with the CIFAR-10 dataset are reported in the following: the first problem is to distinguish classes *airplane* (id: 0) and *horse* (id: 7) while the second is *cat* (id: 3) vs *dog* (id: 5). The cat-vs-dog classification problem is intrinsically difficult since the two classes have many features in common. Moreover, CIFAR-10 have low-resolution images making (sometimes) this classification task not trivial also for human classifiers. On the contrary, airplane-vs-horse is a simple task and thus one should expect that adversarial training does not decrease much natural accuracy. This is confirmed in Fig. 9.6, where it is shown that even in simple settings, adversarial training affects dramatically fine-grained classification.

9.3.2 Unbalanced classification

Although real-world datasets are long-tailed (Deng et al., 2009), most of the experiments and theoretical findings on the accuracy-robustness trade-off in the literature were performed with balanced datasets (Tsipras et al., 2019).

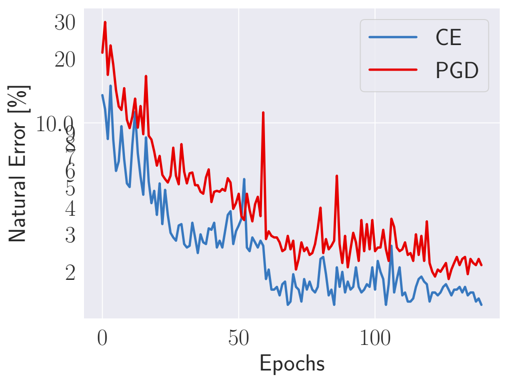
Through an experimental analysis, we show that when classes are unbalanced, adversarial training can have dramatic effects on natural accuracy. For this analysis, the same 2-classes problems of the ablation study reported in subsection 9.3.1 are selected: cat-vs-dog and airplane-vs-horse. Classes are artificially randomly unbalanced such that their ratio is 0.3.

The results of these two experiments are shown in Fig. 9.6: two different considerations are here reported. The first is that when classes are similar, as mentioned above, PGD heavily impacts on the performance with respect to standard training. Instead, for dissimilar classes, the effect is much less pronounced. This a solid argument for supposing that using a single ϵ may be not optimal. The second consideration is that when dataset is unbalanced, PGD further amplifies the difficulty of the classification task. For example, for cat-vs-dog (Fig. 9.6d), in presence of unbalance, the model can't be fit at all.

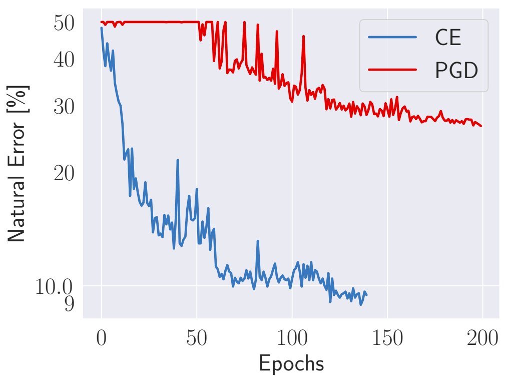
9.3.3 Entropy of softmax outputs

One of the issues of 'standardly' trained network, is that they are over-confident, that is, they tend to predict classes with with high probability even when images are not clear (Guo et al., 2017). Adversarial training can be seen as an implicit regularization and thus it is legitimate to analyze confidence of predictions on robust models. Indeed, in Fig. 9.8 it is shown that another characteristic of adversarial training is reducing confidence of predictions; in fact, the entropy of class logits of the robustly trained network is much higher. This suggests that confidence scores obtained by thresholding the softmax predictions should be changed. Thus, it may seem that robust representations are less discriminative than standard ones. [2] It turns out that this intuition is true and supported by Fig. 9.7. In order to assess the structure of representations, it has been employed t-SNE (Maaten and Hinton, 2008), a technique that allows to visualize high-dimensional data in 2 or 3 dimensions. From Fig. 9.7, it is clear that robust representations are less clustered with respect to natural ones. Each coloured cluster correspond to one particular class.

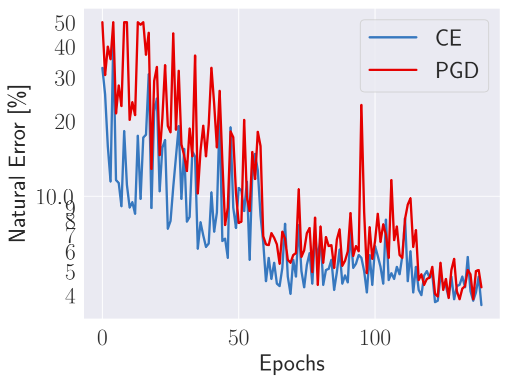
[2]For those who are not used to deep learning language, in this context a representation is the vector (output of the feature extractor) that is feed to the last layer which is a linear classifier.



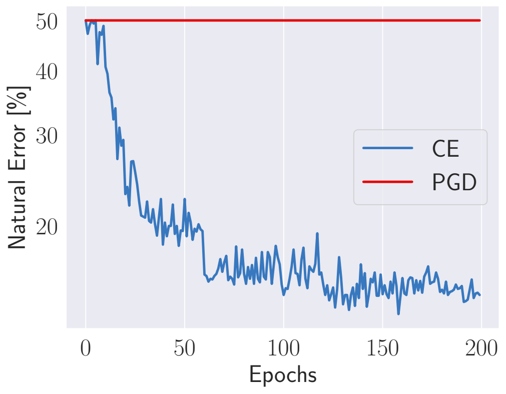
(A) Standard training 0-vs-7.



(B) Standard training 3-vs-5.



(C) Unbalanced 0-vs-7.



(D) Unbalanced 3-vs-5.

FIGURE 9.6: Panel (a) and Panel (b) report validation curves for the problems 0-7 and 3-5, respectively. Panel (c) and Panel (d) are the same curves with a randomly unbalanced dataset. For classes (3) and (5), when the dataset is unbalanced the validation error remains constant to random choice (50%).

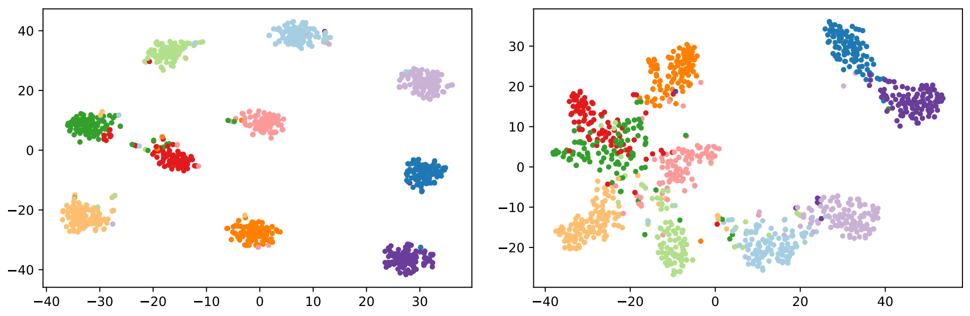
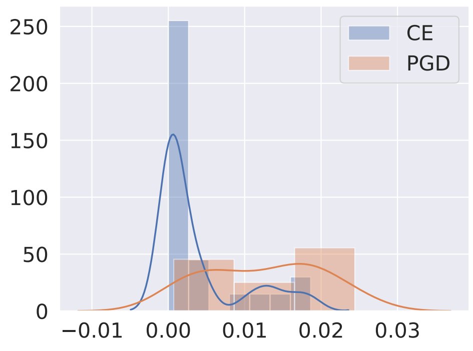
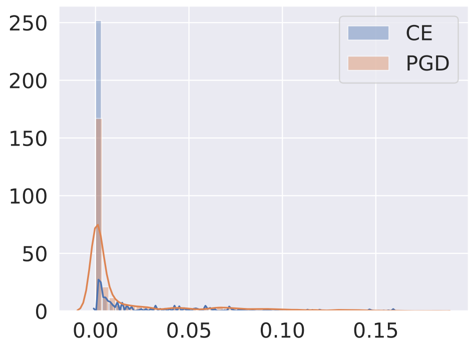


FIGURE 9.7: CIFAR-10. Comparison of t-SNE computed on the representation of standard models (left) and robust models (right). Each coloured cluster correspond to one particular class.



(A) TinyImageNet



(B) CIFAR-10

FIGURE 9.8: Entropy histograms of prediction confidence for W-16-10 with $\epsilon = 8$ of class *airplane*. Robust networks provide more conservative predictions. Adversarial training prevents the network to provide high confidence predictions. This is a consequence of simplifying boundaries as shown in Sec. 9.3.4. Other classes follow the same trend.

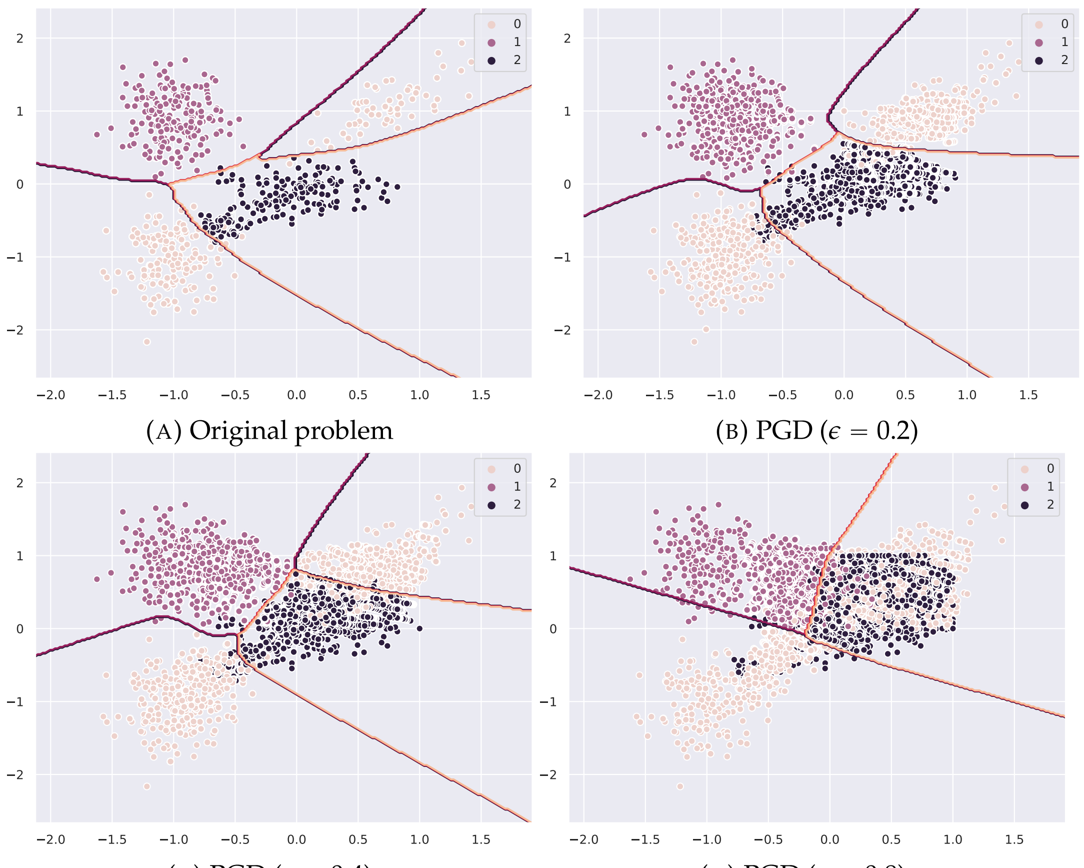
9.3.4 PGD flattens boundaries

In order to better understand the behavior of PGD and also to compare it with WPGD defined in Sec. 9.4, a simple classification problem with three classes is considered. In Fig. 9.9 we show the boundaries for PGD for different ϵ. Fig. 9.10a represents the standard training that achieves almost zero error. As ϵ increases boundaries are more flattened as orthogonal as possible to the gradient direction. We adopt the cost matrix $C = \begin{bmatrix} 0 & 10 & 0.01 \\ 10 & 0 & 1 \\ 0.01 & 1 & 0 \end{bmatrix}$. Related this results, Moosavi-Dezfooli et al., 2018 showed experimentally that the main effect of PGD is to reduce the curvature of boundaries. However, it can be easily shown that even when the curvature is zero, robust training still has an effect. Moreover, it is noticed that gradients are more aligned to the vector which connect two classes. This is due to the "isotropic" effect of PGD which tend to estimate more isotropic distributions. This is in accordance with Tsipras et al., 2019 in which authors observed that gradient on the robust model are more meaningful. This argument is also in accordance to results on fine-grained classification present on this work, suggesting that visually similar are separated by more complex boundaries. Instead, WPGD controls the the regularization of boundaries through the cost matrix: boundaries for couple of classes considered more similar are mostly preserved.

Remark 10. One may find the claim that since visually similar classes are separated by more complex boundaries, it obviously hurts robustness. However, the range of values of ϵ used for robust training are much smaller than the minimal distance between two images in the dataset. Thus, at least in principle, it is not still clear why it is not possible to obtain robustness and accuracy at the same time.

9.4 Wasserstein Projected Gradient Descent

The core motivation of WPGD is that many applications have cost-sensitive errors. For example, in a recognition system of an autonomous car, predicting an image of cat as a dog is not problematic but predicting a human as truck may be very harmful. Ensuring robustness only in prescribed directions have to main advantage of not destroyed accuracy for couple of classes where cost is low. Moreover, PGD uses an



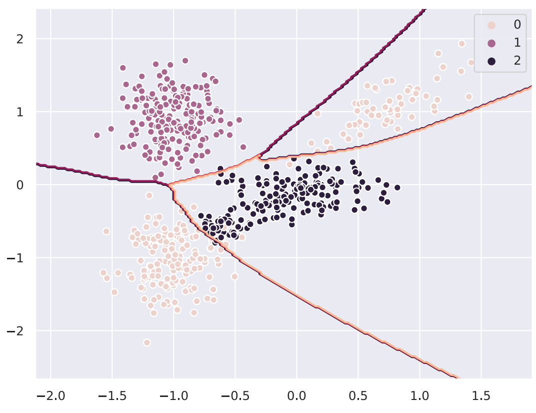
(A) Original problem

(B) PGD ($\epsilon = 0.2$)

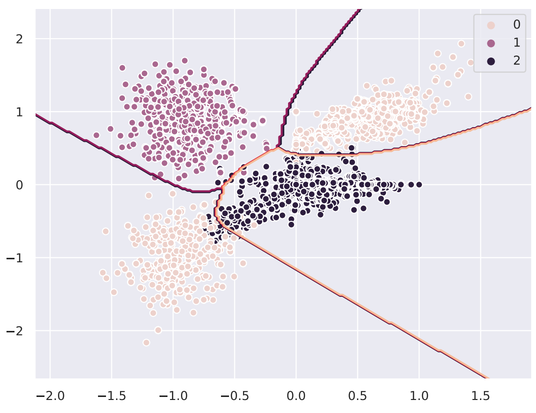
(C) PGD ($\epsilon = 0.4$)

(D) PGD ($\epsilon = 0.8$)

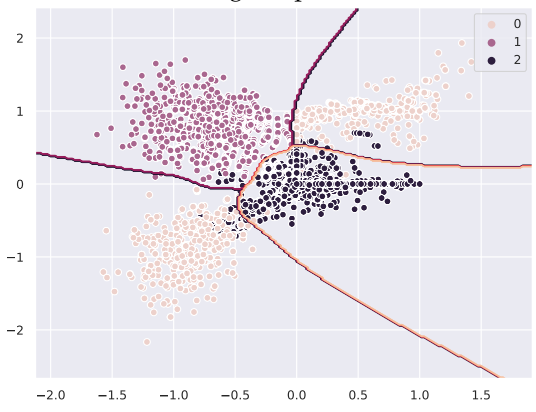
FIGURE 9.9: Effect of robust ℓ_2-training on a simple classification problem. PGD training flattens the boundaries.



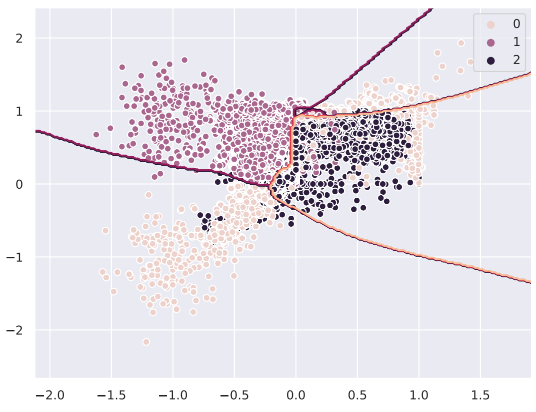
(A) Original problem



(B) WPGD ($\epsilon = 0.2$)



(C) WPGD ($\epsilon = 0.4$)



(D) WPGD ($\epsilon = 0.8$)

FIGURE 9.10: Effect of directional robust ℓ_2-training on a simple classification problem. WPGD flattens the boundaries where the cost is low and preserves them where the cost is high.

isotropic ball and this may be very inefficient. In fact, for certain couples of classes, ϵ may be too high (e.g. cats and dogs that present very similar structures) and for another the opposite.

With this premises, how can we inject a semantic robustness to the problem? A very elegant way is to leverage (discrete) optimal transport tools described in Sec. 7.2.3.

9.4.1 Wasserstein metric and optimal transportation

The first thing we require is the cost that describes the semantic robustness: low-cost couples can be not robust, high-cost couples must be robust. Thus, the label metric is defined as Def. 11. Here, we represent it and the OT loss (7.19) in the context of our problem. The matrix cost between classes is:

Definition 11 (Label metric $C^{\odot p}$). A symmetric positive semi-definite matrix $C \in \mathbb{R}_+^{K\times K}$ defines a pseudo-Riemannian metric on the domain, an entry $C_{k,k'}$ is the cost of transporting unit probability mass from class k to class k'. Note that $C_{k,k} = 0$. The notation $C^{\odot p}$ denotes the element-wise p^{th}-power of C.

In standard classification frameworks, the target is in the so-called hot-encoding form, which simply means that we give probability 1 to the "true" class k and "0" to the other components. Thus loss equal to 0, means that the output $f_\theta(x)$ is 1 in its k-th component and 0 in the others. While (7.19) is for general targets, for our classification problems, this form of target vectors leads to a simplification showed by the following lemma.

Lemma 12 (Closed-form Wasserstein distance). *For any normalized μ, if the target probability distribution ν is a one-hot vector, the Wasserstein distance W_p^p can be computed in closed form and is given by*

$$\mathrm{W}_p^p(\mu,\nu) = \mathrm{C}_{t*}^{\odot p}\ \mu$$

where $t^* = \arg\max_k \nu$. The optimal transport is such that its $(t^*)^{\text{th}}$ column is μ.

The proof of this lemma follows from the observation that the set $\Pi(\mu,\nu)$ is degenerate for one-hot ν, the constraints $\pi^\top \mathbb{1} = \nu$ and $\pi\mathbb{1} = \mu$ force the $(t^*)^{\text{th}}$ column of π to be simply μ. Note that the Wasserstein distance is symmetric and therefore the same statement holds for $\mathrm{W}_p^p(\nu,\mu)$. The previous lemma tells us that one can avoid to run the Sinhorn-Knopp algorithm.
Thus, the loss (7.19) is specialized to our final loss defined in Def. 13.

Definition 13 (Wasserstein Loss). The Wasserstein Loss can now be defined as

$$\ell_{\mathrm{W}}(\theta;\ x) = \mathrm{C}_{t(x)}^{\odot p} f_\theta(x) - \frac{\lambda^{-1}}{\log K}\,\mathrm{H}(f_\theta(x)); \tag{9.2}$$

here $\mathrm{C}_{t(x)}^{\odot p}$ denotes the $t(x)^{\text{th}}$ row of the matrix $C^{\odot p} \in \mathbb{R}_+^{K\times K}$ and $f_\theta(x)$ is the normalized output of the classifier. Note that computing $\ell_{\mathrm{W}}(\theta; x)$ and back-propagating through it has the same computational complexity as standard cross-entropy.

9.4.2 WPGD

The saddle point formulation for the Wasserstein loss (9.2) can be modified to lead to the following definition.

Definition 14 (Robust Wasserstein loss). The Robust Wasserstein loss is defined as

$$\min_{\theta} \mathop{\mathbb{E}}_{X} \ell_{\mathrm{CE}}(x^*; \theta), \quad x^* = \mathop{\arg\max}_{\|x'-x\|_\infty \leqslant \epsilon} \ \ell_{\mathrm{W}}(\theta; x') \tag{9.3}$$

The outer loop remains the same while the inner loop is responsible to find the adversarial example which maximize the Wasserstein loss ℓ_{W}. This implies that at the beginning of training WPGD will prefer directions connecting visually distant classes, such as, cat and truck, preventing to flatten regions between similar classes. It is important to note that during training there is an implicit trade-off between choosing directions suggested by the metrics and gradients directions. In fact, the loss gradient is nothing else that an inner product of the K logit's gradients and the the row k-th row of C.

Imposing an approximation of the real visual metric, helps to efficiently explore the ℓ_∞-ball which, especially for high-dimensional input can be hard to explore, leading to better results. For WPGD experiments we will use that metrics described previously.

9.5 Experiments

This Section provides the experimental findings of the WPGD approach.

9.5.1 Algorithms

We will compare the following four algorithms:

(i) **CE**: This is the standard cross-entropy loss ℓ_{CE} defined in (7.4).

(ii) **PGD**: This is the standard adversarial training algorithm; we solve the saddle-point problem (9.1) with 8 steps in the inner loop to compute the adversarial image.

(iii) **WPGD**: This is the robust Wasserstein loss described in Def. 14 where the inner loop in PGD searches over the adversarial image that maximizes the Wasserstein transport cost. The computational complexity of WPGD is the same as that of PGD. We compared WPGD with three different value of $p = 1, 2.5, 10$.

By W-s-10, we represents the wideresnet architecture with s layers. In order to test robustness we perform 20-steps PGD attacks starting from a random (uniformly sampled) position inside the ℓ_∞ ball of the test image x. All the WPGD experiments are run with the cost matrix provided by the WordNet metric (Miller, 1995).

9.5.2 Directional robustness of WPGD

In Table 9.2 we report the main results of natural training (CE) and robust training (PGD) for CIFAR-10 and TinyImageNet. In Sec. 9.5.3 we report a summary table for quantitative results on directional robustness. Instead, in Fig. 9.11 we show the trade-off arising from WPGD training. As p increases, fine-grained classification is more preserved. In addition to standard accuracy, we compare the characterization of adversarial robustness of PGD ad WPGD. In Fig. 9.12 we show that WPGD-trained networks with a strong metric tend to be more robust between visually distant classes, which supports our claims. For sake of clarity, we report only results for CIFAR-10 and TinyImageNet and W-16-10. Interestingly, WPGD is less robust than PGD for classes *bird* and *airplane*: thus, imposing a metric, even if it is only approximately correct, seems to help to obtain more visually meaningful errors.

C10	**CE**		**PGD**		**Tiny**	**CE**		**PGD**	
	16-10	28-10	16-10	28-10		16-10	28-10	16-10	28-10
NE	4.4	3.9	14.11	13.9	NE	37.7	36.9	55.3	36.9
AE	100	100	34.5	31.25	AE	99.9	100	70.4	70.5

TABLE 9.2: Summary of errors in [%] for W-16-10 and W-28-10, with $\epsilon = 4$ and $k = 20$ under ℓ_∞ perturbations.

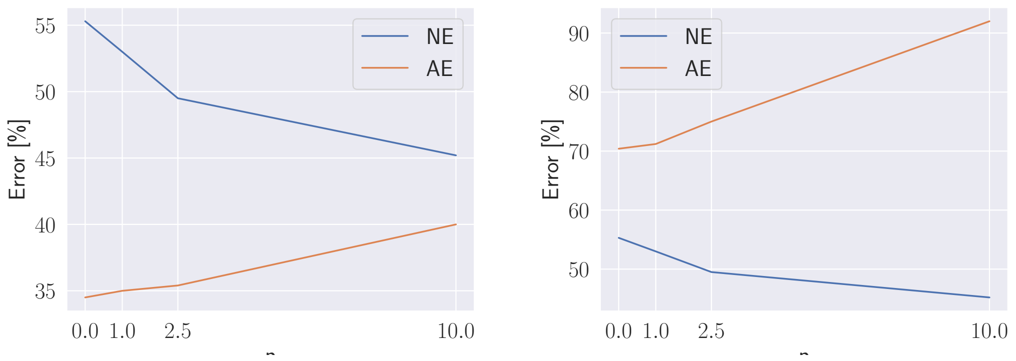
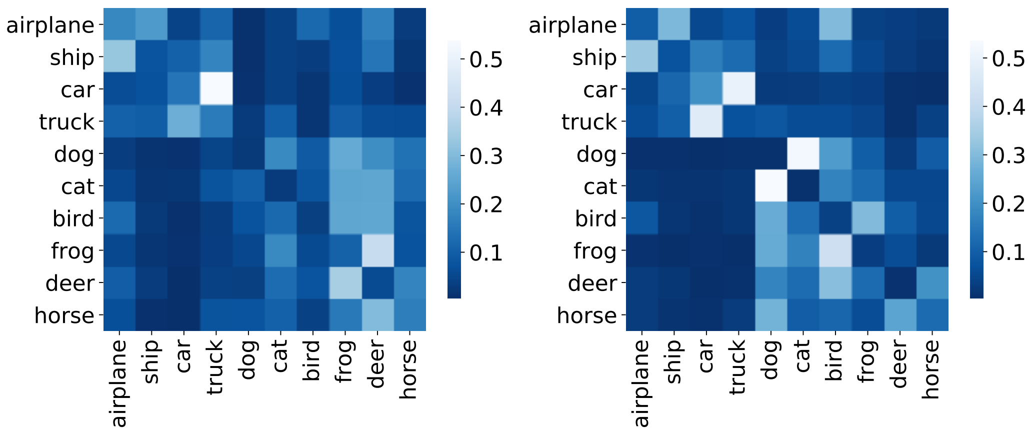


FIGURE 9.11: Accuracy & robustness trade-off: Results for W-16-10 and $\epsilon = 4$, for CIFAR-10 (left) and TinyImageNet (right). Increasing p (x-axis), enables to improve accuracy on fine-grained classification at the price of robustness on pairs of similar classes



(A) **PGD** (B) **WPGD**

FIGURE 9.12: CIFAR-10. Characterization of adversarial robustness for WPGD and PGD defenses. We apply a perturbation of norm $\epsilon = 16$ on W-28-10. WPGD is performed using $p = 2.5$. Fig. 9.12b shows the WPGD obtains directionally robustness: in this case the network is more robust for perturbations between two visually different classes.

9.5.3 Supplementary comparisons for CE, PGD and WPGD

In Fig. 9.13 we report curves plot for PGD and WPGD for CIFAR-10 and TinyImageNet. Moreover, in Table 9.3 we report the summary of weighted robustness score S defined as:

$$S = \sum_{i,j} c_{i,j} m_{i,j} \tag{9.4}$$

where $M = \{m_{ij}\}_{i,j=1}^{K}$ is the adversarial confusion matrix, $C = \{c_{ij}\}_{i,j=1}^{K}$ is the metric of the given dataset. Attacks are computed maximing the loss (9.3), that is considering the worst-case scenario in which the attacker knows the metric. This score weighs more errors in correspondence of high cost. In order to make results legible we set the zero reference to the PGD-trained model. As we can see increasing p, results in reducing the score S, which means that, on average, less similar classes are reached.

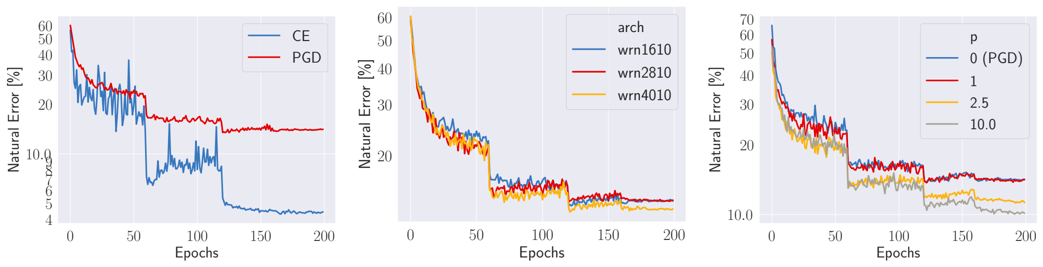


FIGURE 9.13: CIFAR-10. Comparisons: (Left) standard training vs PGD; (Middle) different architectures on robust training; (Right) PGD vs WPGD. WPGD is slightly better in terms of accuracy.

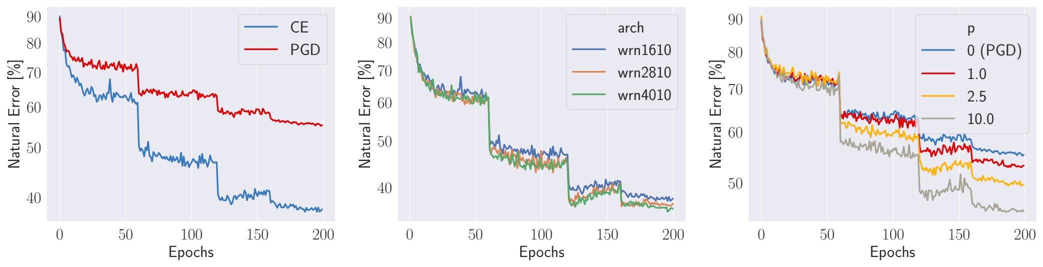


FIGURE 9.14: TinyImageNet. Comparisons: (Left) standard training vs PGD; (Middle) different architectures on robust training; (Right) PGD vs WPGD. WPGD is slightly better in terms of accuracy.

9.6 Conclusions and future work

While the literature on adversarial training is flourishing, profound studies towards understanding its implication and sensitivity to common real-world applications are still lacking. In particular, we focused on applications that are cost-sensitive or the dataset is unbalanced. Moreover, due to an intrinsic trade-off between robustness and accuracy, it is of paramount importance to be able to govern such trade-off when designing and implementing machine and deep learning-based applications where a certain amount of accuracy is required. In lieu of this, in the chapter we made several advances towards understanding better robustness from one side and being able to semantically control it from the other side.

	AE [%]	p	dataset	S
W-16-10	34.53	0.0	CIFAR-10	-0.14
W-16-10	34.62	1.0	CIFAR-10	-0.26
W-16-10	34.98	2.5	CIFAR-10	-0.34
W-16-10	39.76	10.0	CIFAR-10	-0.53
W-28-10	31.24	0.0	CIFAR-10	0.00
W-16-10	70.23	1.0	TinyImageNet	-6.33
W-16-10	73.61	2.5	TinyImageNet	-12.45
W-16-10	92.62	10.0	TinyImageNet	-55.17
W-28-10	69.84	1.0	TinyImageNet	-9.62
W-28-10	69.69	0.0	TinyImageNet	-9.48

TABLE 9.3: Summary of weighted robustness score S defined in (9.4) for $\epsilon = 4$. In order to make results more legible we set the zero reference (for each dataset) to the PGD-trained model with W-28-10. As we can see increasing p, results in reducing the score S, which means that, on average, more similar classes are reached.

In particular, we identified that the accuracy gap in adversarial training comes from the loss of fine-grained classification capabilities in neural networks. This observation motivates the optimal transport formulation: a metric on the label space that measures the distance to the boundary for standard cross-entropy training or, often equivalently, a semantic metric obtained from external data modalities such as WordNet, reduces the search space and makes it easier to discover—and fix—these classes during adversarial training, resulting in an improvement of accuracy at the cost of (directional) robustness. It is conceivable that, although a high-dimensional classifier may always remain vulnerable to adversarial perturbations, it is possible to build robust, real-world systems by incorporating such diverse data. Thus, this work is a first step toward a principled robust training for real-world applications involving artificial intelligence and deep learning.

Future works will regard the study of methodologies or heuristics to systematically control the robustness-accuracy trade-off without the need of tuning ϵ by hyper-parameter tuning. Moreover, another future direction of research is the application of the WPGD approach to other problems like fraud detection and Predictive Maintenance.

Chapter 10

Interpretable representations through minimality

In the context of DL, in the last few years, discriminative models, at least in their classic formulation, have not attracted much attention from the literature. Unfortunately, for various reasons which are out of the scope of this thesis, the community tends to assume a behavior that is not optimal from the research point-of-view: most effort is devoted at creating more and more sophisticated methodologies or variants of very recent methodologies, instead of proceeding with a principled approach that science should require. This led to paradoxical situations where results are not clear because benchmarks are not well-established. For example, in the few-shot learning field, it is not clear what are baseline results to which compare new approaches and the ”variance” of results is so large that a simple careful parameter tuning and very simple approaches can outperform very sophisticated models. This fact (in the context of few shot learning) has been recently pointed out by (Chen et al., 2019) and (Dhillon et al., 2019) where they show that a simple approach or appropriate hyper-parameters tuning can outperform results of a vast variety of complex approaches.

Similarly to few-shot, for discriminative models there has been an exploding literature of complex models (Tan et al., 2018) that try to easily transfer features of models trained on a given task to similar tasks/datasets. This is referred to as *transfer learning*.

In view of this, in the following two chapters we will show that a principled training methodology allows discriminative models to solve different tasks, such as data generation and to transfer better with respect to models obtained by standard training.

In particular, the first part of this chapter is devoted to show that adversarial training and GANs are more similar than what one may think at a first glance. In practice this means that discriminative models can generate new images and potentially can solve other related Computer Vision tasks. Our results are related to a concurrent work (Engstrom et al., 2019)[1] where they present very similar results mostly from a experimental point of view and giving a theoretical argument using simple Gaussian distributions, which are not able to describe the complexity of real image.

The second part has several contributions:

- We define a new concept of minimality that led to a new approach for training robust and interpretable models, which we will refer to them as *minimum energy models* or *natural models*. As we will see, minimality in our framework has several advantages with respect the well-known minimality concept in Information Theory expressed through Information Bottleneck (Tishby, Pereira, and Bialek, 1999; Achille and Soatto, 2018);

[1]The first draft of this thesis has been written before this concurrent work appeared on ArXiv.

- inspired by desirable properties of natural and robust models, we define desiderata for discriminative models in order to consider them interpretable.

10.1 Adversarial training, GANs and Optimal Transport

This section wants to provide a theoretical connection between adversarial training and GANs, which demonstrates that with a single classifier it is possible in practice solve different problems, such as interpolation between images, generations and transportation between classes. Before establishing the connection between OT, AT, and GANs, let us state the following proposition:

Proposition 15. *Let $\mu, \nu \in P(\mathbb{R}^p)$. The $\psi : \mathbb{R}^p \to \mathbb{R}$ be a set of convex potentials. By Brenier's theorem we have that the problem:*

$$\varphi = \arg\min_{\psi} D_f(\nabla\psi_{\#}\mu, \nu) \tag{10.1}$$

is minimized uniquely by φ and $\nabla\varphi$ is the optimal transport map, that is:

$$W_2^2(\mu,\nu) = \int_{\mathcal{X}} \|x - \nabla\varphi(x)\|_2^2 d\mu(x)$$

Proof. By definition, it always holds $D_f(\mu,\nu) \geqslant 0$. By Jensen's inequality we can see that the only solution is given by $\frac{d\mu}{d\mu} = 1$, implying $\mu = \nu$. On the other hand, by Brenier's theorem we know that, if φ is convex, it is the only minimizer of the Monge-Kantorovich problem with quadratic cost. ■

Proposition 15 can be generalized to other strictly convex costs with a slightly different transport map (Villani, 2008). In solving the problem in Proposition 15 we encounter two main problems: the first is that we require a convex map and deep learning models are not convex neither in the parameters nor in the input. We will come back to the relation between convexity and ResNets in Sec. 10.7. In any case, in order to solve the optimal transport problem (7.9) it is not necessary for the potential φ to be convex. In that case, the map is not expressed by the gradient of convex potential but by (7.12). The other problem is that we are assuming the we have a good estimate of D_f. However, this is not true and it will be clarified in the next two subsections.

10.1.1 Standard training

Finding a classifier $c_\theta(x) = \arg\max_{t\in\mathcal{T}} f_\theta(t|x)$ which distinguishes μ and ν is equivalent to correctly estimate a divergence (or distance) D between them.

While a generative model has the objective of estimating $p_\theta(x)$ or $p_\theta(x \mid t)$ for conditional generators, the objective of a classifier is to provide a parametric estimate of $p(t \mid x)$. How are generative models related to classifiers?
Standard training using cross-entropy loss amounts to solve:

$$\mathcal{L}_{ST} = \min_\theta \int_{x\in X} \varphi(x) d\mu(x) + \int_{y\in Y} \psi(y) d\nu(y) \tag{10.2}$$

with $\varphi(x) = -\log f_\theta(t = -1 \mid x)$, $\psi(y) = -\log f_\theta(t = 1 \mid y)$ and $f_\theta(t = i \mid y)$, $i = \{-1, 1\}$, are the result of the softmax function on the network (not normalized) output. Remembering that the final total mass is conserved, we have:

$$\mathcal{L}_{ST} = \min_{\theta} \int_{x \in X} -\log f_\theta(t = -1 \mid x) d\mu(x) + \int_{y \in Y} \log\left(1 - f_\theta(t = -1 \mid y)\right) d\nu(y) \tag{10.3}$$

As we can see it has a very similar form of (7.27). Thus, from the previous arguments, we can see that the estimation of a classifier can be seen equivalent to estimate the divergence/metric between two distributions. Throughout this thesis we will consider a binary classification problem between distributions μ and ν. In general, the optimization objective to be solved for a classifier is:

$$\mathcal{L}_{cl} = \min_{\varphi, \psi} \int_{x \in \mathcal{X}} \varphi(x) d\mu(x) + \int_{y \in \mathcal{Y}} \psi(y) d\nu(y) \tag{10.4}$$

$$\text{s.t. } F(\varphi(x), \psi(y)) \in \Omega(x, y), \forall x \in \mathcal{X}, y \in \mathcal{Y}, \tag{10.5}$$

where φ and ψ are potentials relative to the respective distributions, F is a function, that, together with the set Ω defines the constraint and the domains depending on the divergence used. For example, the estimation of a classifier using the cross-entropy loss can be seen equivalent to estimate (up to an additive constant) the D_{JS} divergence between two distributions μ and ν. Thus, a good classifier that minimizes loss (10.4) should be also able to estimate its relative divergence. Now let us consider the problem (10.2). If we seek a distribution ρ maximizing the classification loss, we should have that:

$$\min_{\rho} \mathcal{L}_{ST}(\rho)$$

is solved by $\rho \equiv \nu$, since $D_{JS}(\rho \| \nu) \geqslant 0$ and we have $D_{JS}(\rho \| \nu) > 0$ for $\rho \neq \nu$. However, with standard classifiers it is well-known that this is not true. In fact, we can find small perturbations from examples generated by μ (that we assume still belong to the same distribution) that are classified as belonging to ν. In order words, we can find points where the estimation of the corresponding divergence is not valid. In order to reduce this problem, one may add isotropic noise to the input. However, since input data is high-dimensional, its effect is very mild. Fortunately, there exists a more efficient way to inject noise, as we will see in the following section.

10.1.2 Adversarial training

Adversarial Training (AT) was originally introduced to make classification models robust. However, we will show that it has a direct connection with GANs. AT amounts to solve the following mini-max problem:

$$\mathcal{L}_{rob} = \min_{\theta} \int_{\mathcal{X}} \varphi(x^*) d\mu(x) + \int_{\mathcal{Y}} \psi(y^*) d\nu(y) \tag{10.6}$$

where:

$$x^\star = \arg\max_{x' \in \mathcal{B}_\epsilon(x)} \varphi(x')$$

$$y^\star = \arg\max_{y' \in \mathcal{B}_\epsilon(y)} \psi(y')$$

and $\mathcal{B}_\epsilon(x) = \{y : \|y - x\|_2 \leqslant \epsilon\}$. Here we maintained ψ and φ general. With cross-entropy loss, we have $\varphi(x) = -\log f_\theta(t = -1 \mid x)$ and $\psi(y) = \log\left[1 - f_\theta(t = -1 \mid x)\right]$ as in (10.2).

The main idea under AT is that a small perturbation of a sample drawn from a class μ still belongs to μ. That is, a perturbed image representing a cat is still a cat. In particular, from a training point-of-view, we have the following intuition: fixed the model's parameters θ, the adversary tries to find the bounded directional perturbation $d\mu$ such that $D_{JS}(\mu + d\mu, \nu)$ decreases most. Then, the model is updated to take into account of this mismatch. This results in an adversarial game (similar to GANs) where the adversary has more and more difficulty to find an effective perturbation. At convergence, we expect that the estimation of D_{JS} is (at least locally) much more stable and smoother, and that gradient direction points to the other class. If we consider the sample-based version of D_{KL}[2] (7.24):

$$\hat{D}_{KL}(\mu, \nu) = 1 + \max_g \left\{ \frac{1}{n_x} \sum_i g(x_i) - \frac{1}{n_y} \sum_j e^{g(y_j)} \right\}$$

and g_0 is the true maximizer, the inner maximization of (10.6) is equivalent to find a sample (or a batch of samples) $x^\star$ from the distribution μ such that:

$$\begin{aligned} x^\star &= \operatorname*{arg\,max}_{x' \in \mathcal{B}_\epsilon(x)} \left\{ \hat{D}_{KL}\left(\mu(x') \| \nu\right) - \hat{D}^{\hat{g}}_{KL}\left(\mu(x') \| \nu\right) \right\} \\ &= \operatorname*{arg\,max}_{x' \in \mathcal{B}_\epsilon(x)} \frac{1}{n_x} \sum_i \left[g_0(x'_i) - \hat{g}(x'_i) \right] \end{aligned}$$

In other words, with a general f-divergence the adversary aims at finding a region where the current $\hat{g}$ does not approximate well the *true* divergence $\hat{D}_f\left(\mu(x') \| \nu\right) \approx \hat{D}_f\left(\mu(x) \| \nu\right)$. When the divergence is symmetric, like JSD, the same automatically applies for the direction $\nu \to \mu$.

10.1.3 Computation of the adversarial transport map

Once the robust classification model is trained and the estimate of the f-divergence $D_f(\mu \| \nu)$ is good enough we can transport $x \sim \mu$ to a $y \sim \nu$ solving the usual problem problem:

$$x^\star = T_\epsilon(x) = \operatorname*{arg\,max}_{x' \in \mathcal{B}_\epsilon(x)} \varphi(x') \tag{10.7}$$

with the difference that ϵ is now set to bigger values that enable to transport enough mass from μ to ν. The solution of this problem is different from the solution x of $\max_{x \in X} -c(x, y) + \varphi(x)$, which is the transport map on the OT problem. This map is not optimal in general. In fact, it may exists a point $x^{\star\star} \in \nu$ such that $\|x - x^{\star\star}\|_2 < \|x - T_\epsilon(x)\|_2$, where we made explicit that x^* depends on ϵ. In other words, while for OT the transport map consists of a maximization that entails a trade-off between cost and potential, for AT, the the adversary is allowed to spend all the budget inside the ball. Hence, it is optimal in the sense that, fixed a budget, it extracts the best profit. In the generation task, this is not a problem since we are only interesting in getting a realist image describing a given class. Instead, if we want to solve to optimal

[2]We use the KL divergence for simplicity and the argument does not change using the JS divergence.

transportation, it is necessary to tune an ϵ optimal for the task/dataset or for the single image. In order to ensure optimality one should solve:

$$x^\star = \underset{x' \in \mathcal{B}_{\epsilon^\star}(x)}{\arg\max}\, \varphi(x') \qquad \text{s.t.} \quad x^\star \sim \nu$$

where $\epsilon^\star$ is the minimal value possible such that $x^\star \sim \nu$.

10.1.4 Relaxation of adversarial training

In order to gain more insights on the effects of adversarial training, we consider the relaxation of its inner loop:

$$\phi_\lambda(x) = \max_{x'} \left\{ \varphi(x') - \lambda c(x', x) \right\} \tag{10.8}$$

where the penalty constant λ fixed. Contrary to AT, its relaxed version has a soft constraint that allows to trade off between φ and the cost c.
The transportation map becomes:

$$x^\star = \underset{x'}{\arg\max} \left\{ \varphi(x') - \lambda c(x', x) \right\} \tag{10.9}$$

and we immediately recognize that $x^* = T_\lambda(x)$ coincides with the transport map defined by (7.16). This is not a case. In fact, Blanchet and Murthy, 2019 showed that the surrogate surrogate ϕ is intimately connected with optimal transport. This connection is provided by the following proposition, where we consider $\mathcal{X} \equiv \mathcal{Y}$.

Proposition 16. *Let $\varphi : \mathcal{X} \to \mathbb{R}$ and $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ be continuous. Consider the surrogate version (10.8). For any distribution ν and any $\delta \geqslant 0$, we have*

$$\max_{\mu : W_c(\mu,\nu) \leqslant \delta} \mathop{\mathbb{E}}_{x \sim \mu} \varphi(x) = \min_{\lambda \geqslant 0} \left\{ \lambda\delta + \mathop{\mathbb{E}}_{y \in \nu} \phi_\lambda(y) \right\} \tag{10.10}$$

where W_c is the Wasserstein metric with cost c. Moreover, for $\lambda \geqslant 0$, we have:

$$\max_{\mu} \left\{ \mathop{\mathbb{E}}_{x \sim \mu} \varphi(x) - \lambda W_c(\mu, \nu) \right\} = \mathop{\mathbb{E}}_{\nu} \phi_\lambda. \tag{10.11}$$

The previous proposition shows that the expectation over a distribution ν of the surrogate (10.8), is equivalent to find the transport from the distribution ν to μ.

Moreover, we would like to understand the role of $\phi(x)$ from a geometrical point of view. The following proposition shows that using the relaxed version of AT corresponds to an implicit regularization similar to manifold regularization (Belkin, Niyogi, and Sindhwani, 2006).

Proposition 17 (Implicit regularization). *Let $\phi(x)$ defined as in (10.8) and assume that its optimization problem is strongly-concave. This can be guaranteed for example when $\nabla_x \varphi(x)$ is L-Lipschitz and $\lambda > L$, for some L. Moreover, assume a quadratic cost $c(z) = \|z\|_2^2$ Then, the problem (10.9) is equivalent to minimize the regularized loss (for the class k):*

$$\mathcal{L}(x;\theta)_k = \varphi(x;\theta)_k + \frac{1}{2} \|\nabla_x \varphi_k^\top(x) \nabla_x \varphi_k(x)\|_{\bar{F}^{-1}} + O\left(\frac{1}{\lambda^2}\right) \tag{10.12}$$

where $\bar{F} = F - \lambda I$ and F is the hessian of the negative log-likelihood $F(x) = \nabla_x^2 \log p(x)$.

The proof is given in Appendix A.1. Proposition 17 shows that AT provides a data-dependent regularizer which depends on the local geometry defined by $\bar{F}^{-1}$ and λ controls the smoothness scale. In order to give a more direct insight, we consider a simple linear classifier with a logistic regression loss:

$$\varphi_k(x) = -\log f(x)_k, \qquad f(x)_k = \frac{\exp \theta_k^t x}{\sum_{j=1}^K \exp \theta_j^t x}$$

of the k-th class. As we can easily see, $H = 0$ and the loss gradient is:

$$\nabla_x \varphi_k(x) = -\theta_k + \sum_{j=1}^K f(x)_j \theta_j$$

which implies the, with a linear classification model (A.1) becomes:

$$\mathcal{L}(x;\theta)_k = \varphi(x;\theta)_k + \frac{1}{2\lambda} \|\theta_k - \sum_{j=1}^K f(x)_j \theta_j\|_2^2.$$

The second term in the rhs acts as a regularizer that penalizes the distance of the barycenter model to the linear classifier for the true class k.

10.1.5 Role of ϵ

Setting a good a value of ϵ is not trivial for several reasons: there are pairs of classes that are very similar and others that are very different suggesting that a unique value of ϵ may not be optimal. Moreover, it is not even clear what it is purpose of setting ϵ. From a semantic point-of-view, ϵ has to be set to a value that allows to not discard discriminative features of classes. The more ϵ, the more the structure of image manifold is destroyed towards an isotropic shape. Moreover, the effect of increasing ϵ, is to increase the overlap between the supports of different classes. In fact, in low-dimensional manifolds, like images, the probability of finding a classifier that separates the two classes tends to 1, since classes have disjoint supports with high probability. Actually, this is one of the reasons for which standard training does not estimate well the divergence between classes. A clear example is the cat-vs-dog classification task: standard models can be very accurate but robust models even with a small ϵ have a much lower accuracy. This is due to the fact the manifold of cats is very close to the one of dogs. From a more mathematical point-of-view, adversarial training softens the divergence used that usually is the D_{JS}. In fact, as already mentioned above, a common strategy to solve the problem of disjoint manifolds is to inject noise, usually Gaussian. However, since that data lies in a low-dimensional manifold, but input is high-dimensional, the effect of the noise is very low and thus a large bandwidth is required to obtain a smoothing effect. Instead, adversarial training is optimal since at every iteration of training excites the most informative directions, given by the gradient flow. In Fig. 10.1 we show an example of adversarial training on the landscape of a simple classification of two Gaussians distributions: while the standard model have an uninformative landscape that saturates very quickly, the robust model present a softer and smoother landscape.

This argument leads us to two interesting ideas. The first is, when training GANS, to adopt a robust discriminator instead of the standard discriminator. This will allow to stabilize gradients and making them more meaningful.

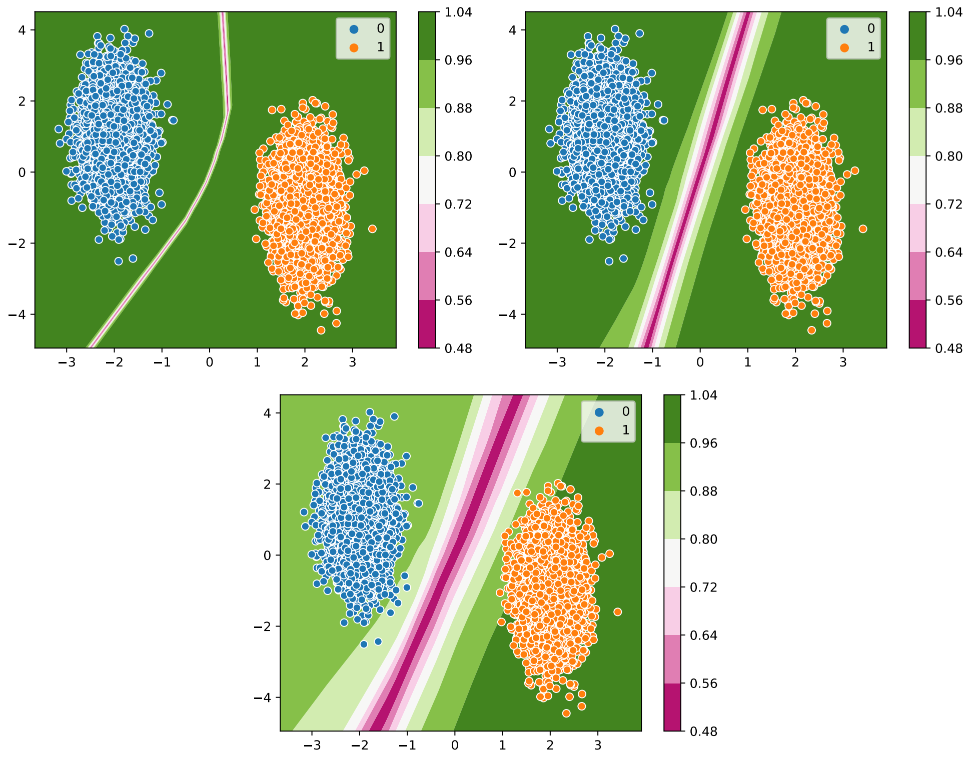


FIGURE 10.1: Softening effect of adversarial training on two Gaussians of form $\mu = \mathcal{N}(m, \Sigma)$ and $\nu = \mathcal{N}(-m, \Sigma)$. The colormap represents the output probability of the two classes. (Top-Left): Standard training. (Top-Right): AT with $\epsilon = 1.8$. (Bottom): AT with $\epsilon = 2$.

The second idea comes from the following point: while the divergence is softened, AT requires the model to be stable around a ball defined with a given norm. The training procedure imposes that the probability of a given class inside this ball is fixed and constant, going to possibly affect directions that should be preserved and not perturbed. One solution to solve this problem is to anneal ϵ while training. We will investigate this ideas in a future work.

10.1.6 Differences between AT and GANs

A natural question to ask is what are the differences between AT and GANs from the from the transportation point-of-view.
The main differences are:

- AT applies the same adversarial game of GAN but the adversary is bounded to $\mathcal{B}_\epsilon(x)$, while for GANs there is no bound and moreover, the AT adversary is optimal because it follows the gradient directions. Instead, for GANs the alternate update of discriminator and generator may lead to uninformative gradients leading to mode collapse;

- For GANs, the source distribution is a noise distribution, usually an isotropic Gaussian, while for AT the source distribution is already a real distribution corresponding to a class;

- this in turn causes a different transport map. In fact, GANs estimate the transport maps T_ξ directly using a parametric model while for AT it is $T(x) = \arg\max_y \varphi(y) - c(x, y)$ which is the c-transform (7.10). In words, AT estimates the Kantorovich potential. Estimating the potential has an advantage in the context of DL: deep networks compute differentiable maps however the true transport may be not so smooth. While the generation network in GANs is differentiable (almost everywhere) by construction, a differentiable (almost everywhere) potential can have a discontinuous transport map. For example, let us consider a convex potential φ solving the optimal transport problem (7.8). Then, for the Brenier's theorem we have that the optimal transport is the gradient of this convex potential $T(x) = \nabla\varphi(x)$. Now assume $\varphi(x) = |x|$, which can be easily approximated by a NN. Its gradient is sign x that is highly discontinuous. Moreover, another advantage of AT with respect to GANs is that training is stable without all tricks required for training generative networks.

- the optimization objective: in fact, in the classification problem there is no explicit requirement to match the input data, that is, to give high probability $p(y \mid x)$ to real images. In fact, in Sec. 10.4.6 we will show that features of real images and generated images are not aligned;
- AT justifies linear interpolation in the representation space. This is not true for GANs and VAEs since they have to explore taking into account of the sampling distribution. In fact, in the generation of images, especially in GANs, many tricks are applied to overcome this and other issues (Brock, Donahue, and Simonyan, 2018) We will clarify it in Sec. 10.1.7.

10.1.7 Linear interpolation

With generative models such as GAN and VAE, the characterization of the latent space is an open problem and recent literature attempted to perform inference exploiting linear interpolation. For example, Agustsson et al. (2017) finds a measure-preserving map which enables to explore the latent space through linear interpolation. We will see that with models that solve the optimal transport problem, the latent space is optimally explored using linear interpolation. In fact, as a natural consequence of displacement interpolation in the input space, the optimal displacement in the space of latent representations is linear. The next well-known proposition gives the tools we need.

Proposition 18 (Straight lines). *If $c(x, y)$ is a convex cost function in $\mathbb{R}^p$, then:*

$$\inf\left\{\int_0^1 c(\dot{z}_\tau)d\tau; \quad z_0 = x, z_1 = y\right\} = c(y - x)$$

Moreover, if c is strictly convex, the infimum is uniquely given by straight lines:

$$z_t = x + \tau(y - x)$$

Proof. By Jensen's inequality we have:

$$\left\{\int_0^1 c(\dot{z}_\tau)d\tau; \quad z_0 = x, z_1 = y\right\} \geqslant c\left(\int_0^1 \dot{z}_\tau d\tau\right) = c(y - x)$$

The infimum depends only on the displacement $y - x$ which means that optimal paths are straight lines. The strict convexity guarantees that the solution is unique. ■

The previous proposition tells us that, in a Euclidean space, the optimal transport is composed by straight lines. For example:

$$C\left[(\dot{z}_\tau)\right] = \int_0^1 \|\dot{z}_\tau\|^2 d\tau \qquad \Rightarrow \quad c(x,y) = \|x-y\|^2$$

$$C\left[(\dot{z}_\tau)\right] = \int_0^1 \|\dot{z}_\tau\| d\tau \qquad \Rightarrow \quad c(x,y) = \|x-y\|$$

We recall that $r(\cdot)$ is the representation network before the last linear layer $Ar(\cdot) + b$. Thus, we can write $\varphi(x) = w^\top r(x) + b$. Now, it is immediate that if φ solves the the optimal transport for a given cost $c(x,y)$, then the linear model (A, b) solves the optimal transport for $c(r(x), r(y))$. Since the model is linear then the optimal transport in the representation space must be linear:

$$r(z_\tau) = (1-\tau)r(x) + \tau r(y),$$

where $0 \leqslant \tau \leqslant 1$. We will provide an example of image interpolation in Sec. 10.4.3.

10.2 Natural models

Although adversarial training can help solve the optimal transportation problem, setting the optimal ε is challenging and requires to be tailored to the particular dataset. Proposition 18 gives us an alternative formulation of the classification problem (10.20). Informally, our aim is to map (or transport) data lying in a complex manifold to a global Euclidean space.

Let ℓ_{CE} be the usual cross-entropy loss $\ell\left(f_\theta(x), t\right) = -\log f_\theta(x)$. The constraint to be satisfied in order to solve the optimal transportation problem is that features r_t are linear along the *unconstrained* gradient flow that is:

$$r(z_\tau) = (1-\tau)\cdot r(x) + \tau \cdot r(y) \text{ s.t. } \begin{cases} \dot{z_\tau} & = \nabla \ell_{CE}(z_\tau) \\ z_0 & = x \end{cases} \tag{10.13}$$

where x, y and z_τ are the initial, final and intermediate (at time τ) images, respectively. In order to provide a fair comparison with adversarial training we use the $\|\cdot\|_2$ norm.

Thus, the discretized flow amounts to compute T steps:

$$z_{k+1} = z_k + \eta \frac{\nabla \ell_{CE}(z_k)}{\|\nabla \ell_{CE}(z_k)\|_2}$$

where η is the step size.We would like to remark that there is no constraint on interpolates to fall in a predefined ball.

Unfortunately, obtaining a global linear output is very hard and moreover, there is no guarantee that it is even possible with a finite amount of parameters. However, requiring local linearity can be sufficient for our task. In order to satisfy such constraint we impose the following loss ℓ_{flow}:

$$\ell_{flow}(x;\theta) = \|r_\theta(z_\tau) - [(1-\tau)\cdot r_\theta(x) + \tau \cdot r_\theta(y)]\|^2 \tag{10.14}$$

or the normalized loss:

$$\ell^n_{flow}(x;\theta) = \frac{\|r_\theta(z_\tau) - [(1-\tau)\cdot r_\theta(x) + \tau \cdot r_\theta(y)]\|^2}{\|r_\theta(x)\|^2} \tag{10.15}$$

We empirically found (10.14) to be easier to set up, thus from now on we will only consider (10.14). The total ”Lagrangian” loss becomes:

$$\mathcal{L}_{NAT} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{flow} = \tag{10.16}$$
$$= \mathop{\mathbb{E}}_{x,t\in\mathcal{D}} \left[\ell\left(f_\theta(x), t\right) + \lambda \ell_{flow}(x)\right] \tag{10.17}$$

where λ has to be set big enough in order to guarantee the constraint (10.13) is satisfied.

We refer to this procedure as *Natural Training* (NT). Minimality here is expressed by the triplet (λ,η,T). These three are not independent of each other: for example, in first approximation, computing $n \cdot T$ steps with a given η is equivalent to perform T steps with step size $n \cdot \eta$. Instead, λ determines how strict the linearity constraint is.

10.3 How does minimality manifest itself? A comparison with Information Bottleneck

In the recent years, literature studied DL from an information-theoretic point of view (Achille and Soatto, 2018) through the concept of Information Bottleneck (IB). Before showing the math under IB, it is worth giving an informal idea on the problem under exam. Assume we want to solve a particular task, that in our case is a classification task. In order to do so, we have to find a representation of the input data that is sufficient for the task via the minimization of an empirical risk. This is what standard training does, where the risk is expressed through the cross-entropy loss. However, among all the sufficient representations, we would like the one that is invariant to parts of the data that are not important for the task.

More formally, we want to find a random variable r satisfying the following conditions:

1. r is a **representation** of x; that is, its distribution depends only on x
2. r is **sufficient** for the task t, that is $I(x;t) = I(r;t)$
3. among all random variables satisfying these requirements, the mutual information $I(x;r)$ is **minimal**. This means that r discards all variability in the data that is not relevant to the task.

It is easy to see that the aforementioned conditions are equivalent to finding a distribution $p(r|x)$ which solves the optimization problem

$$\begin{aligned} \text{minimize} \quad & I(x;r) \\ \text{s.t.} \quad & H(t|r) = H(t|x). \end{aligned}$$

where H denotes the entropy and I the mutual information. In general, this minimization is very difficult. A natural relaxation of this problem is the usual Lagrangian relaxation, that is this context is called Information Bottleneck (IB), was introduced by Tishby, Pereira, and Bialek, 1999:

$$\mathcal{L} = H(t|r) + \beta I(x;r). \tag{10.18}$$

where β is a positive constant that controls the trade-off between sufficiency (the performance on the task, i.e. the cross-entropy loss, as measured by the first term) and minimality (the complexity of the representation, measured by the second term).

As we can see, the way in which we arrived to the "Lagrangian" loss (10.16) is very similar to the rational under IB: β and λ are intimately related and they represent the strength of the relative constraints. In fact, both their original formulations are burdensome: our formulation would like global linearity, IB wants minimum complexity.

Now two questions remain unanswered:

1. what are the differences between IB and NT?
2. How does minimality manifest itself in our approach?

IB and NT, not only differ conceptually but also technically. First, IB requires a stochastic representation to be well-defined, while NT can be defined generally with measures: in principle, neither it requires probability distributions nor the network to be stochastic. Moreover, IB-minimality means that the representation r stores the least amount of information possible about x. Instead, NT-minimality means that the model (through its representations r) spends the least energy possible passing from one distribution the other. This is more natural from a physics-oriented point-of-view. Actually, NT-minimality does not require that information about the input is discarded, and r can be possibly an invertible map of the input. We will show it in the experiments. In the case of invertible map, the mutual information between representations r and input x, would be $+\infty$. Now, coming back to the second question: how do we see the effect of NT-minimality? First we need to compare the effect of β and λ: when $\beta \to +\infty$, we constrain the model to remove almost all the complexity. For example, for the reconstruction task (that is, generating images on VAEs), this amounts to pushing towards a solution that only maintains low-order statistics of images, removing all the details. When $\lambda \to +\infty$, we requires more and more local linearity, where "local" tends to "global" as $\eta \cdot T \to +\infty$. Thus, where this is very hard to obtain (e.g. cat-vs-dog), the model discards features that do not allow for a linear flow. We remark that, at least in theory, there is no requirement of reducing complexity in natural training. As an effect of this procedure (valid also for adversarial training) is that the high-probability examples are the minimal examples. Sec. 10.4.7 will better clarify this fact with examples. Moreover, this is confirmed also by Sec. 10.4.6 where we will see that features (representations) of real and generated images are distinguishable.

10.4 Experiments

In this section, we will provide experimental results that support our previous argumentation about the connection between AT and GANs through Transport theory and results for NT. In order to produce fair experiments (NT-vs-AT), we set hyper-parameters in such a way to obtain a similar natural accuracy for both the approaches with the same number of PGD steps. Since the results of our approach proposed Sec. 10.2 are qualitatively similar to the ones obtained via AT, we will present them only when they are different. In this context, we found in all the experiments that less iterations are required for natural models with respect to robust models. For simplicity, we set all the hyper-parameters (e.g. number of iteration of the optimization) in common to the same values. Moreover, we found that transportation is more natural, especially when a smaller ϵ is used. Fig. 10.5 are examples of transportation using natural models.

We propose four experiments: image transportation, image interpolation, image generation and image inversion. All the experimental setup for the following experiments is reported in Appendix B.

Moreover, in Sec. 10.4.7 we will show the effects on increasing λ, i.e. enforcing more minimality.

10.4.1 Image transportation

Image transportation simply consists on starting from a source class μ and maximizing the probability of a target class ν by solving (10.7):

$$x^{\star} = T_{\epsilon}(x) = \arg\max_{x' \in \mathcal{B}_{\epsilon}(x)} \varphi(x')$$

Here we will show results only for CIFAR-10 Fig. 10.2 and RestrictedImageNet Fig. 10.3. While for CIFAR-10 transportation leads to images that reasonably belong to the target class, for RestrictedImageNet results are less satisfactory. While there may be several reasons for this result for RestrictedImageNet, here we identify one of them, that, as will see, is the reason for unsatisfactory on data generation experiment. Inspecting the gradient flow of the transport optimization problem, one can see that the loss converges to 0 after few steps, implying that also the norm of the gradient approaches zero. This may be the manifestation of the masking effect also noted while training GANs (Arjovsky, Chintala, and Bottou, 2017). In view of this, instead of minimizing the loss for a given target class, we maximize the loss for all the other classes but the target. We refer to it as reverse optimization. Results for image transportation on RestrictedImagenet are reported in Fig. 10.4 and shows that transport images are more realistically belonging to the target classes.

The main difference we noticed between AT and NT is that the latter obtains more satisfactory results in the transportation problem. Fig. 10.5 is an example of transportation for NT-trained models. We have not been able to obtain the same quality of results for AT.

10.4.2 Image generation

Since now we have seen how to transport one image of class μ to an image of class ν. Is it is natural to ask whether is possible to generate images similarly to GAN and VAE. A trivial solution to generate a new sample for a given class k is to fit a simple distribution such as a normal distribution $\mathcal{N}(\bar{x}_k, \Sigma_k)$ where $\bar{x}_k$ and Σ_k are

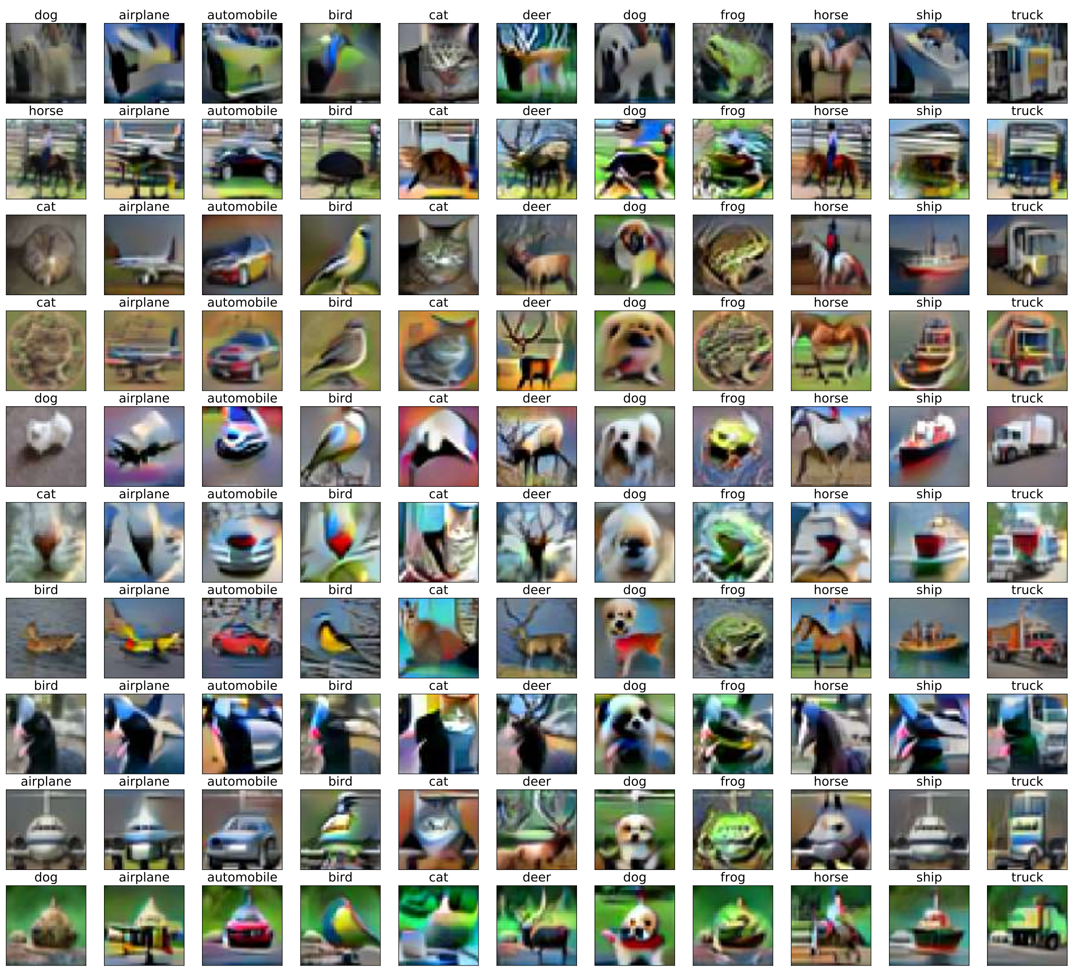


FIGURE 10.2: **CIFAR-10**. Examples of image transportation using ResNet-18. Left column: Original images with their respective classes. From second to last column: transport image with respect the given target class. Hyper-parameters are reported in Appendix B.1.4.

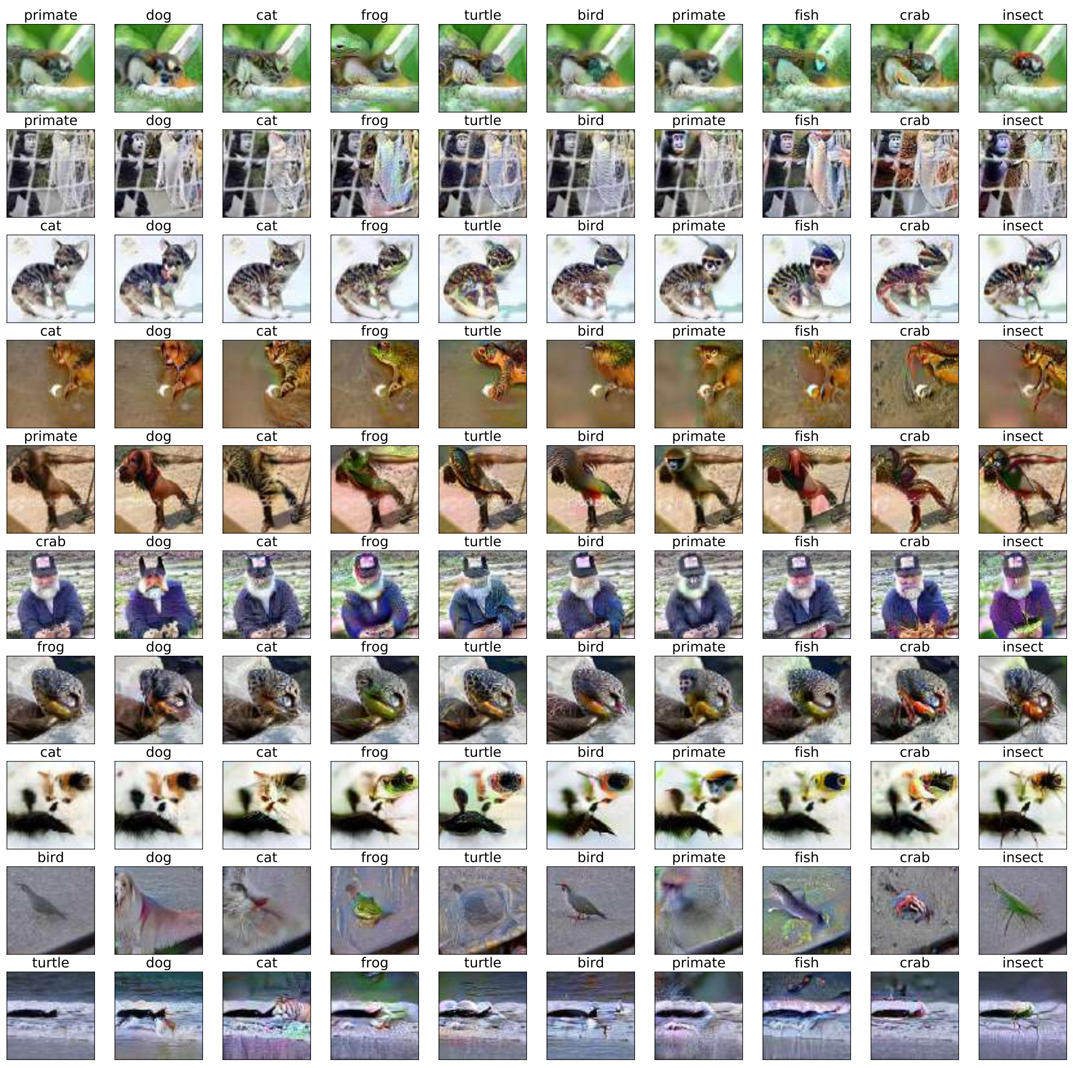


FIGURE 10.3: **RestrictedImageNet**. Examples of image transportation using ResNet-18. Left column: Original images with their respective classes. From second to last column: transport image with respect the given target class. Hyper-parameters are reported in Appendix B.1.4.

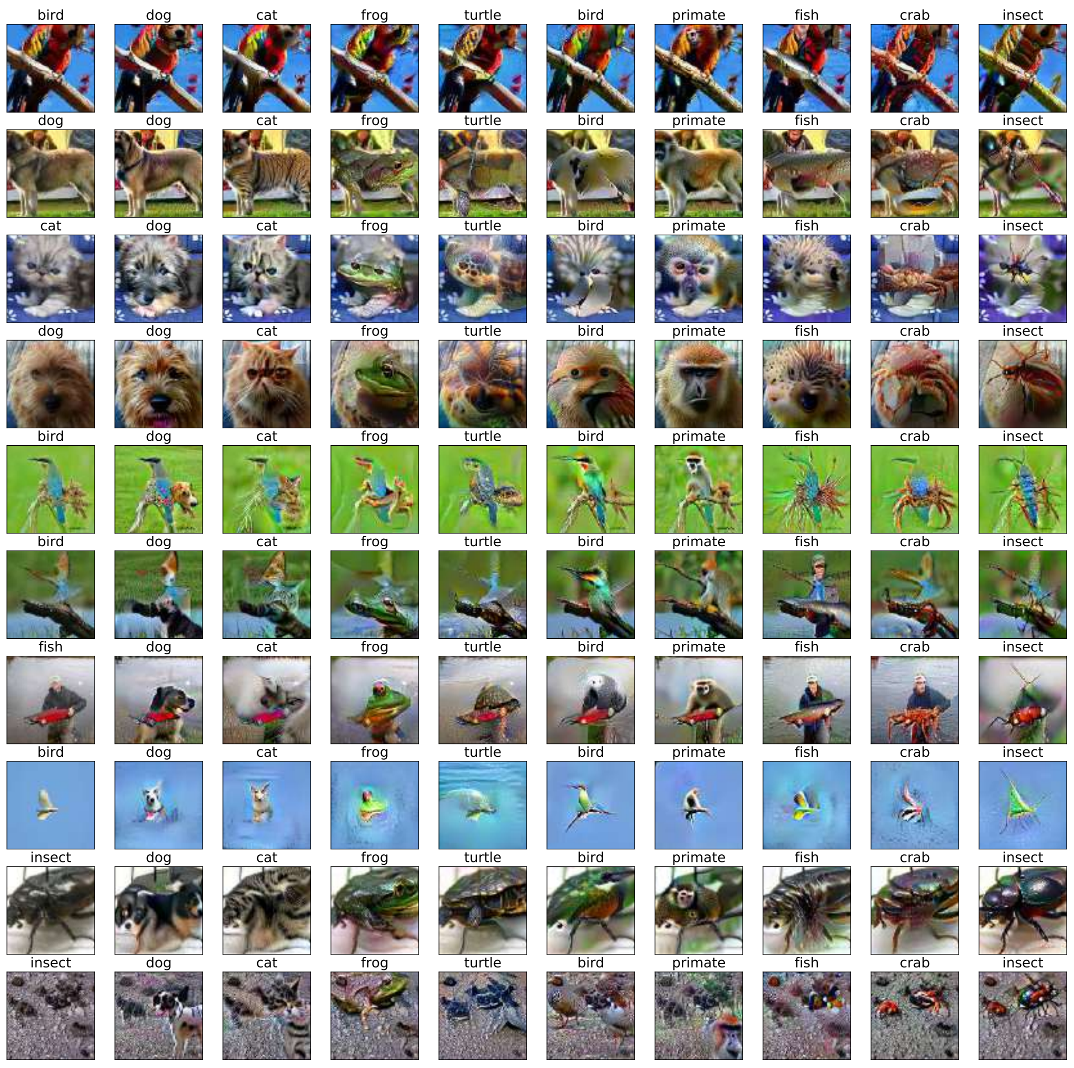


FIGURE 10.4: **RestrictedImageNet**. Examples of image transportation using ResNet-18. Left column: Original images with their respective classes. From second to last column: transport image with respect the given target class. Hyper-parameters are reported in Appendix B.1.4.

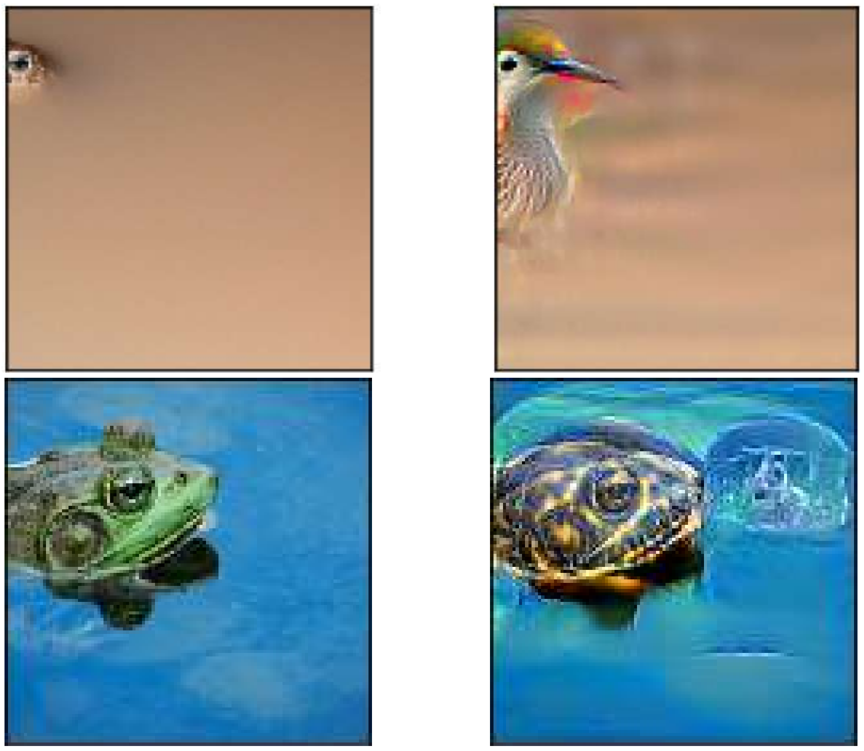


FIGURE 10.5: Optimal transport example with natural models. Starting from a seed image (left), we find the optimal transport to the target class (right). (Top) Unknown to bird, (bottom) frog to turtle.

the empirical Euclidean mean and variance of class samples. As a second step, the generated sample is found by maximizing the probability of class k:

$$x = \arg\max_y f_\theta^k(y \mid y_0), \quad y_0 \sim \mathcal{N}(\bar{x}_k, \Sigma_k)$$

However, data distribution is not normal and discards higher-order statistics. A better approach would to fit a simple distribution on the latent space R such as $\mathcal{N}(\bar{r}_k, S_k)$ and then compute the new sample x by "inversion":

$$x = \arg\min_y \|r(y) - \tilde{r}\|_2, \quad \tilde{r} \sim \mathcal{N}(\bar{r}_k, S_k)$$

Here, for sake of simplicity, we only employ the first method since it already provides reasonable results. In Fig. 10.6 and Fig. 10.7 we present the results for CIFAR-10 and RestrictedImageNet, respectively.
Similarly to the image transportation experiment, the masking effect causes images to be low-quality. In Fig. 10.8 one can see that applying the same 'reverse' procedure discussed in Sec. 10.4.1, quality of images increase.

10.4.3 Image interpolation

In Fig. 10.9 we show that, linear interpolation in the representation space, leads to a semantically meaningful interpolation in the image space.

We compute the interpolates in the simplest manner. Let x and y be the source and target images, respectively. At time τ, the optimizer starts from the seed image

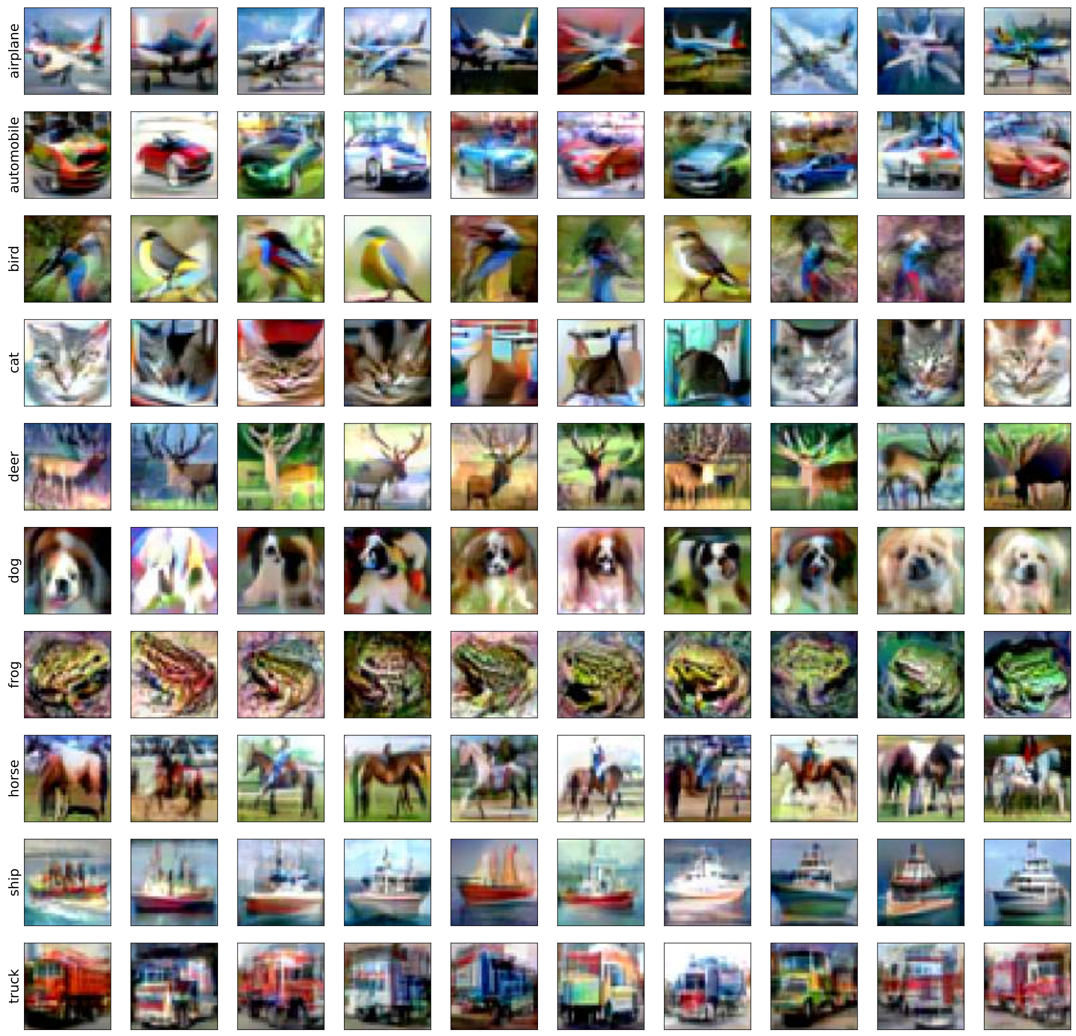


FIGURE 10.6: **CIFAR-10**. Examples of generate images. Hyper-parameters are given in Appendix B.1.6.

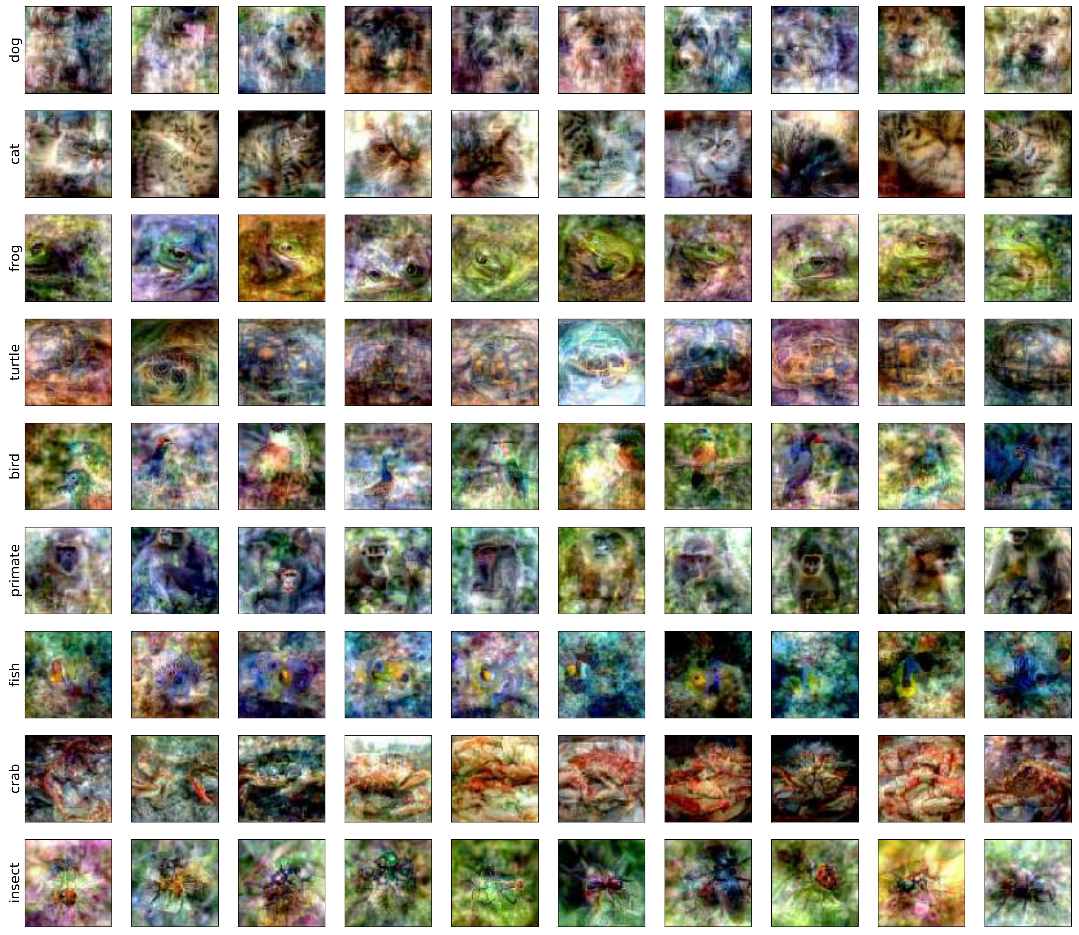


FIGURE 10.7: **RestrictedImageNet**. Examples of generate images. Hyper-parameters are given in Appendix B.1.6.

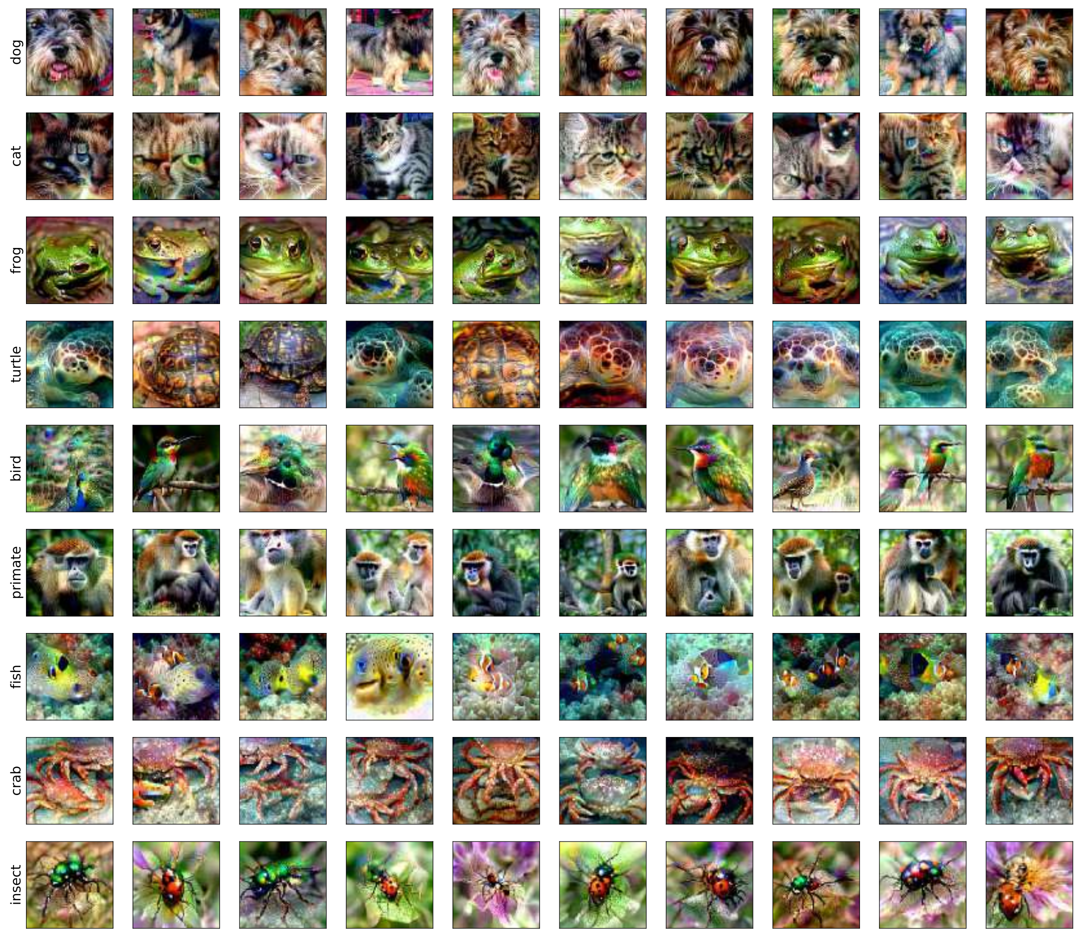


FIGURE 10.8: **RestrictedImageNet**. Examples of generate images with reverse optimization. Hyper-parameters are given in Appendix B.1.6.

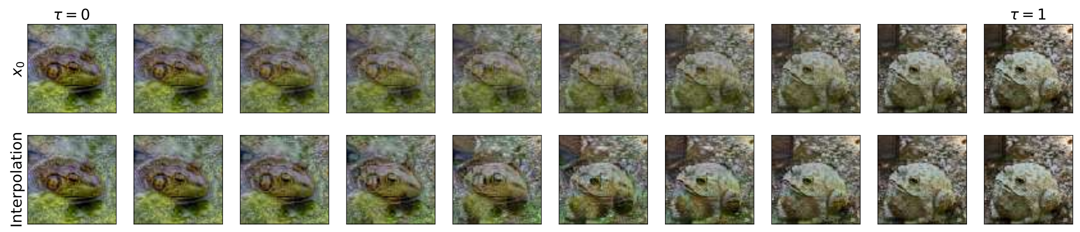


FIGURE 10.9: **RestrictedImageNet**. Example of image interpolation. Differently from linear image interpolation, images obtained by matching the linear interpolation of representations are more semantically meaningful. While linear image interpolation overlaps in transparency the two images, the optimal interpolation of the robust/natural models keeps the image cleaner. We suggest to zoom the image.

$x_\tau = (1-\tau)x + \tau y$, and finds the image z_τ that matches r_τ, i.e.

$$\bar{z}_\tau = \arg\min_z \|r(z) - \bar{r}\|^2$$

where $\bar{r} = (1-\tau)r(x) + \tau r(y)$.

10.4.4 Image inversion

Are representations invertible? First, let us define invertibility. Assume that an input x has a representation $\bar{r}$. Let $x^\star$ be the result of the following optimization problem:

$$x^\star = \arg\min_y \|r(y) - \bar{r}\|^2$$

where $r(y)$ the usual output (representation of x) of the feature extraction network. We say that the model is invertible if $\|x^\star - x\| \leqslant \gamma$, for a small gamma. In other words, we would that the (pseudo) inversion $x^\star$ looks like x. Fig. 10.10 shows that the invertibility assumption is approximately satisfied.

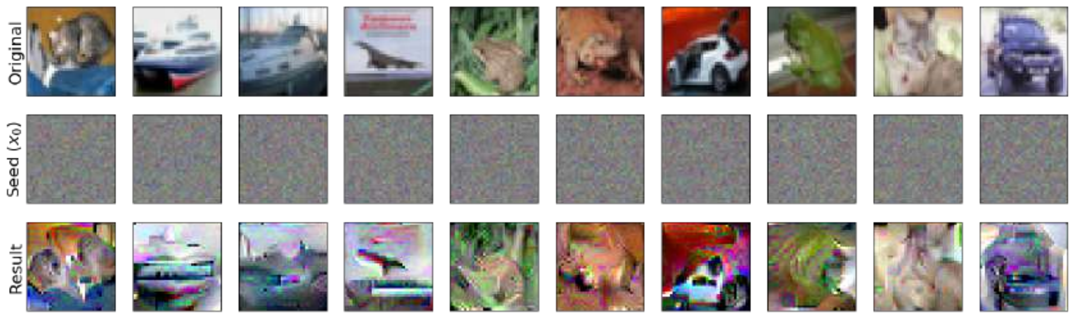


FIGURE 10.10: **CIFAR-10**. ResNet-50. Representation are approximately invertible.

10.4.5 Transferability

All the previous results suggest that features in robust and natural models are more transferable. In fact, let us consider the representation output as the new input features to the linear classifier. Indeed, since linear classifier is optimal (by construction) for a task (e.g. CIFAR-10), it optimally "linearly" interpolates among classes, and features are "invertible" (in a weak sense), we should aspect that the feature extractor is sufficient to linearly classify another dataset (via a new linear classifier). In order to test this hypothesis we run a simple experiment: we retrained the final linear layer of a pre-trained robust/natural and standard networks for a new task. For this experiment, we used a ResNet-50 trained on CIFAR-10. The new tasks are CIFAR-100, CINIC-10 (Darlow et al., 2018) and SVHN (Netzer et al., 2011). Results on Table 10.1 supports the above argument but for CINIC-10. The latter is an extended version of CIFAR-10, with additional ImageNet images. Moreover, some of the training CIFAR-10 images are present on the CINIC-10 test test. This is why results for CINIC-10 are not in favour of the robust model. These results suggest that the standard network is an expert for a particular task while robust/natural models are more similar to a meta-learner. Moreover, we noticed that simply updating BatchNorm (BN) statistics of the feature extractor can provide a not-trivial improvement on the performance. However, for sake of clarity, for all datasets but CIFAR-100, we only report results obtained with the feature extractor completely freezed.

CIFAR-10 →	CIFAR-100	CINIC-10	SVHN
Standard	70.0 (64.0)	**20.5**	63.0
Robust	**61.0** (**55.0**)	24.5	**41.3**

TABLE 10.1: Transferability of a robust models trained on CIFAR-10 on new tasks. Transferability is measured by the percentage top-1 error: the lower, the better. The standard network performs better on CINIC-10 since this is an extended version of CIFAR-10 and, moreover, some of the training CIFAR-10 images are present on the CINIC-10 test test. Results in parenthesis are obtained by updating the BN statistics: simply "registering" the statistics allows to reduce the error of 6% on CIFAR-100.

10.4.6 Registration

Although we have seen that robust training helps to obtain transportability there is no guarantee that the generated images belong to the actual distribution. In fact, from a network-like point of view there may be no difference for a "stylized" cat and a real cat. In order to experimentally verify it, we trained a *linear* discriminator $D_\xi(z)$ whose objective is classify real images versus generated images through their representations. It is worth noting while discriminators in GANs receive in input an image x, $D_\xi(z)$ receives the image representation $r(x)$. What we found for both the datasets that the discriminator achieves almost zero error confirming that features of real and generated images are different.

Is it is legitimate to suppose that the robust network discarded details and features that are present in real images. An approximate way to check this is to train a robust discriminator real-vs-generated and then trying to computed the usual transport map with respect this new 2-classes discriminator. Fig. 10.11 shows images generated by transporting seed images to the class 'real'. As we can see, although some details are present, images are not realistic, especially in how parts of the animals are connected together. Interestingly, when trying to transport to the class 'fake' (Fig. 10.12), images contain parts of different classes. This experiment preliminary suggests that high-order features of real images are discarded. In fact, as we already remarked, from the classifier point-of-view there is no need to maintain the spatial order between patches. However, this is not a definitive results: an ablation study is required to distinguish the roles of the discriminator (here we only employed a linear classifier) and the robust network. This will be subject of future work.

10.4.7 Effects of NT-minimality

In order to assess the effect of minimality, we trained a ResNet-18 on CIFAR-10 with $\lambda = 10^6$, almost 10 times bigger than the standard λ. We observed that accuracy of the model decreases of 6% passing from 10% error to 16%. In fact, this a natural consequence of the fact that we imposed too much linearity for the given dataset and network. A more interesting effect showing clearly the role of the imposed constraint is given in Fig. 10.13: minimality reveals itself by requiring that maximum probability is given to only to "stylised" examples, removed of uninformative details.

10.5 Extensions

In this section, we briefly present two extensions of our approach to tackle semi-supervised and unsupervised scenarios. A detailed study of these problems is left as a future work.

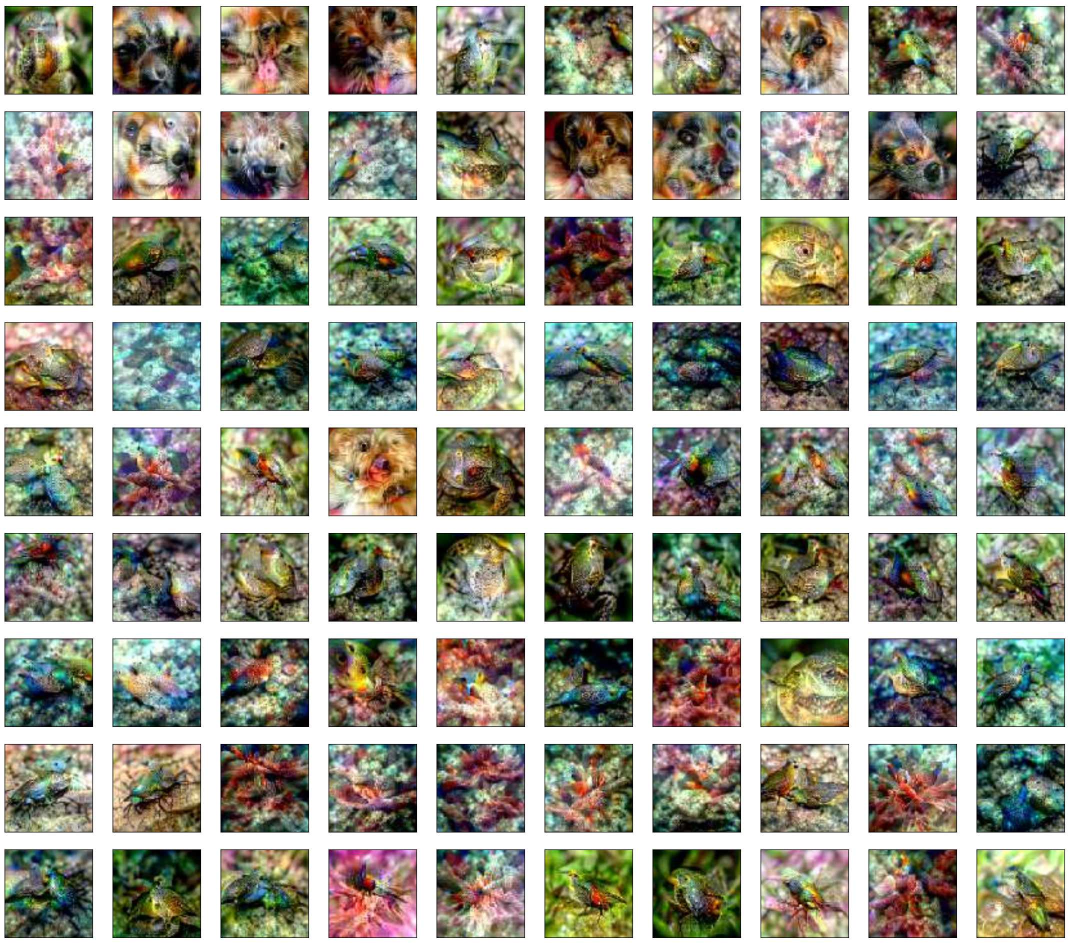


FIGURE 10.11: RestrictedImageNet. Images generated by maximizing the probablity of the class 'real' inside a 2-ball of radius $\epsilon = 40$. Some classes, especially *dogs* present a destructured parts.

10.5.1 Semi-supervised training

The constraint (10.13) is independent of the label t and it is only a property of the representations. Thus, it is natural to exploit information from unlabelled examples, possibly coming from different probability distributions. In fact, Raghunathan et al., 2019; Stanforth, Fawzi, Kohli, et al., 2019 recently showed that Unsupervised Adversarial Training helps to improve generalization and robustness, by requiring stability of the output (unknown) prediction. Let $X = X_s \cup X_u$ be the union of supervised (s) and unsupervised (u) examples, then the total loss becomes:

$$\begin{aligned}\mathcal{L}_{semi} &= \mathcal{L}_{CE} + \lambda \mathcal{L}_{flow} = \\ &= \mathop{\mathbb{E}}_{x_s, t \in \mathcal{D}} \ell\left(f_\theta(x_s), t\right) + \lambda \mathop{\mathbb{E}}_{x \in X} \ell_{flow}(x)\end{aligned}$$

In this case $\ell_{flow}(x)$ is evaluated along the flow which maximizes the f-divergence between a virtual $f_\theta(x) : \mathbb{R}^n \to \mathbb{R}$ target distribution and the current $f(x)$:

$$x^* = \arg\max_{x'} D_f\left[f(x) \| f(x')\right].$$

This approach has two advantages:

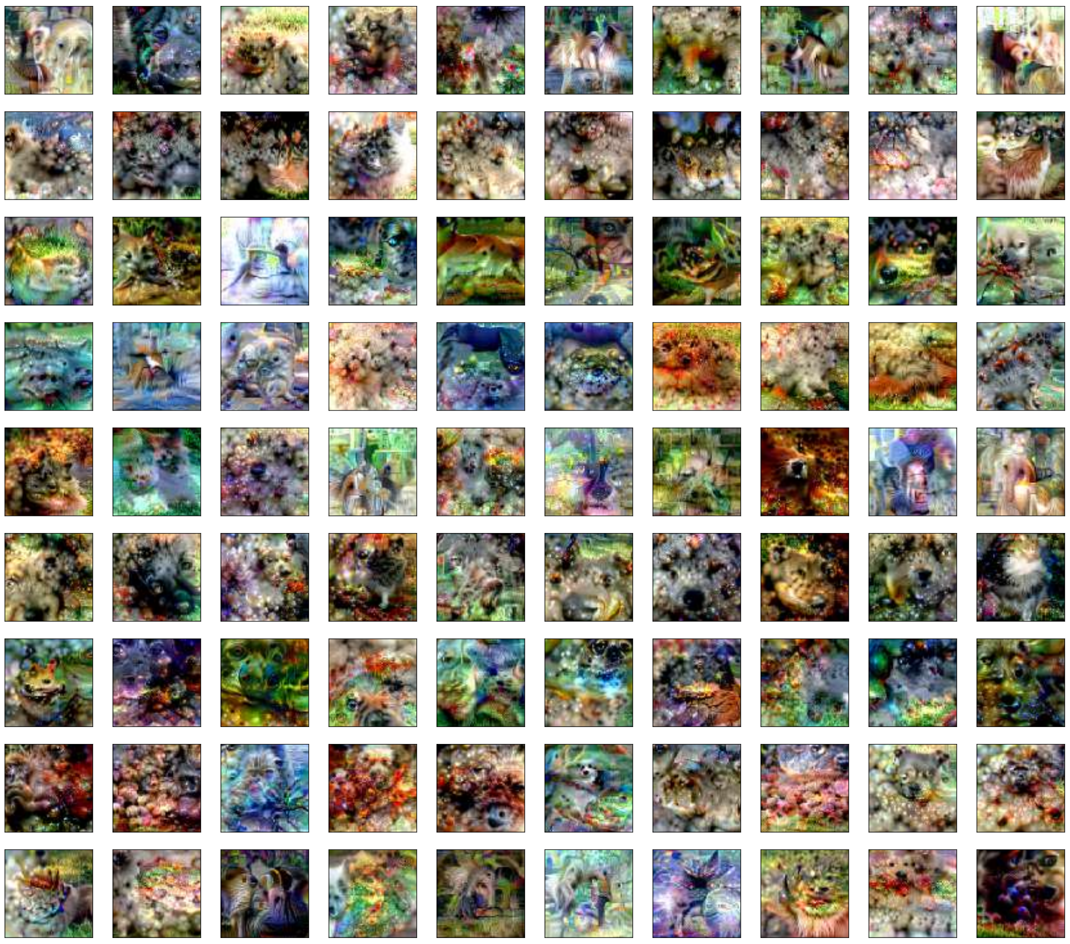


FIGURE 10.12: RestrictedImageNet. Images generated by maximizing the probability of the class 'fake' inside a 2-ball of radius $\epsilon = 40$. As we can see they are result of a mix of features of different classes.

- by jointly exploiting supervised training and regularity on unsupervised data, it is possible to obtain smoother and more meaningful representations;
- it is possible to reduce the size of the labelled dataset, since the unsupervised loss acts as an implicit regularizer preventing overfitting.

10.5.2 Unsupervised training with self-representations

At this point one may push the previous problem to the limit, that is, to a complete unsupervised setting. From one side, it seems ill-posed since only applying the linearity constraint would lead to degenerate and singular solutions, e.g. a constant output. On the other side, some of the issues characterising supervised problems are not present, as for example, the noisy and ambiguous labels. Moreover, in the common benchmark datasets, the number of effective classes is always bigger: for example, say CIFAR-10, has more than 10 classes, because there are different types of cars, cats, frogs, etc. As a consequence, a classifier can perfectly solve the task removing information that allow to distinguish the effective classes.

At the limit, we may think training images belong to different classes (e.g. 100 images means 100 classes). In this case we could resort again to (10.16) where the new dataset $\tilde{\mathcal{D}}$ is $\tilde{\mathcal{D}} = \{x_i, t_i\}_{i=1}^{N}, t_i \neq t_j$ when $i \neq j$. This approach does not scale well with

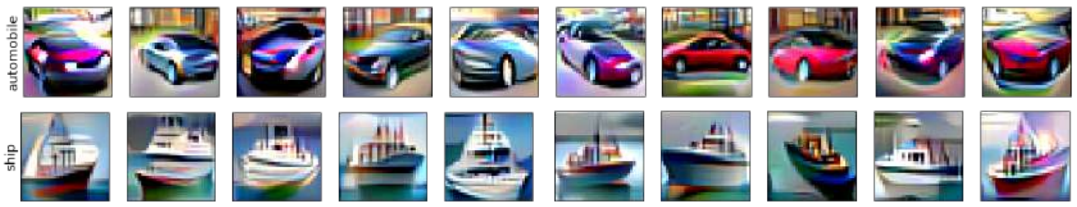


FIGURE 10.13: **CIFAR-10**. For a highly regularized model (to achieve minimality), the most-likely samples are "stylized" and have only the necessary features.

the number of images N as it scales linearly with it. A simple solution sampling from the dataset a large number of images e.g. 1000 and then classify them. Without the linearity constraint, the model would overfit the data (Zhang et al., 2016) since only one sample per class is given. But, enforcing the linearity constraint as a regulariser, this overfitting phenomenon would not happen.

Thus, if we consider the new dataset $\mathcal{D}_u = \{u_i, t_i\}_{i=1}^{M}$, where $t \in \{0, M-1\}$, we have:

$$\begin{aligned}\mathcal{L}_{semi} &= \mathcal{L}_{CE} + \lambda\mathcal{L}_{flow} = \\ &= \mathop{\mathbb{E}}_{x,t\in\mathcal{D}_u} \ell\left(f_\theta(x), t\right) + \lambda \mathop{\mathbb{E}}_{x\in\mathcal{D}_u} \ell_{flow}(x)\end{aligned}$$

In Fig. 10.14 we provide the results of a preliminary experiment[3] showing that learnt features are almost invertible.

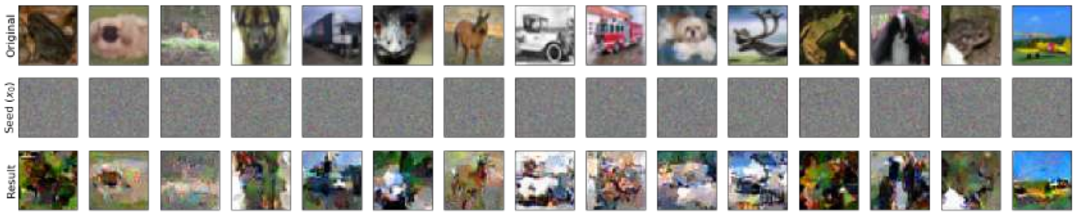


FIGURE 10.14: **CIFAR-10**. ResNet-18. Inversion of representations of the unsupervised model.

10.6 Desiderata for classification models

How do we define interpretability? What are the properties a model should have to be considered interpretable? Informally, we can say a model is interpretable if its behavior is easily understandable and correct from the human point of view. Of course, this is too vague. Alvarez-Melis and Jaakkola, 2018 proposed three desiderata for interpretability:

1. Explicitness/Intelligibility: Are the explanations immediate and understandable?
2. Faithfulness: Are relevance scores indicative of "true" importance?
3. Stability: How consistent are the explanations for similar/neighboring examples?

[3]Hyper-parameters were set without a careful tuning. Hence, we expect that results have a large margin of improvement.

They introduce a structure called Self-Explaining neural networks (SENNs) which are based on the idea that models locally should behave like linear models and that output decision should depend of few sparse concepts. The drawback of their approach is that they require to minimize multiple losses and to set a particular structure to ensure the given constraints.

In view of the nice properties of robust and natural models, we propose four desiderata that do not require a change of structure of existing models and that can be precisely defined: **Stability**, **Transportability**, **Minimality** and **Transferability**. These properties are not independent. For example, we experimentally showed that models that are stable, minimal and transportable are also more transferable.

10.6.1 Stability

The concept of stability is not new and informally says that a model output does not change much under a small perturbation applied to the input. More formally:

Definition 19. We say a classification model f_θ is (δ, ϵ)-stable under a norm p at point x, if, given a small dx, with $\|dx\|_p \leqslant \epsilon$, it holds: $|\ell(f_\theta(x + dx)) - \ell(f_\theta(x))| \leqslant \delta$.

We would like to point out that ϵ and δ depend on x, which means that we require that the model is locally linear but the bandwidth of 'locally' depends on the point x. Do robust models trained with adversarial training (PGD) satisfies stability? Partially. Indeed, robust models 'guarantee' to be stable under a ball $\mathcal{B}_\epsilon$ for a given norm $\|\cdot\|_p$. However, e is fixed, that is the ball both isotropic and fixed for all the points $x \in \mathcal{X}$. Thus, in principle, the couple (δ, ϵ) may not optimal for a given point x.

10.6.2 Transportability

Informally, we say a classification model is *transportable* if, starting from the distribution of one class, the gradient path obtained by maximizing the loss leads to the actual distribution of another class. More formally this means that the solution of the problem:

$$z^\star(x) = \arg\max_{z \in \mathcal{B}_\epsilon(x)} \varphi_\theta(z) \tag{10.19}$$

must satisfy the condition $z^\star(x) \sim \nu$.

The reason under this requirement is simply that the gradient path must be meaningful and the an increment of the loss must correspond to a true visually meaningful change towards the target (or nearest) class. It is well-known that standardly trained classification models don't satisfy this property even locally: in fact, there exist small perturbations along the gradient path that change the predicted class even if the true class is unchanged. The perturbed examples are said to be *adversarial*. We would like to remark that: (i) we do not only require that the gradient path leads to a plausible image but to its more plausible counterpart, and (ii) this definition only requires a transport map T such that $T_\# \mu = \nu$ without any optimality constraint, that is φ may be not optimal.

10.6.3 Minimality

As we already discussed before, from a physical point of view, we may think the most natural (optimal) model is the one at minimum energy. But how do we define the concept of minimality? In fact, there are many different ways of defining this property in the context of Machine Learning. Good examples are all the forms of parameter or information-theoretically based regularization (Achille and Soatto, 2018). However, here we prefer a more natural and dynamical definition of minimum energy:

Definition 20. Consider the problem of distinguishing two classes represented by the distributions $\mu : \mathcal{X} \to \mathbb{R}$ and $\mu : \mathcal{Y} \to \mathbb{R}$ respectively. With an abuse of notation, let f_θ a classifier solving the optimization problem

$$\min_{\theta} \mathop{\mathbb{E}}_{x,t \in \mathcal{D}} \ell\left(f_\theta(x), t\right)$$

We define the minimum energy model as the one minimizing the following quantity:

$$\begin{aligned} &\mathop{\mathbb{E}}_{x \sim \mu} \int_0^\infty \frac{1}{2} \|\dot{z}(\tau, x)\|^p \, d\tau \\ &\text{s.t. } T_\# \mu = \nu \end{aligned} \tag{10.20}$$

where T is defined by the end point of the gradient flow of the loss function.

In words, Def. 20 reads: the minimum energy discriminator is the one for which starting from the distribution μ, gives the shortest path to the distribution ν, fixed the cost $\|\cdot\|^p$. For example, if we are trying to classify cats and dogs, this definition requires that the flow given by the loss ℓ results in a minimum energy path, causing a meaningful and natural interpolation between these classes. Under mild assumptions, among all the feasible classifiers, only one is optimal.

Minimality as defined here is strictly connected to optimal transport. In fact, it coincides with the dynamical formulation of optimal transport proposed by Benamou and Brenier, 2000.

10.6.4 Transferability

One of the main reserach topics in Machine Learning is model *transferability*. Informally, a model is transferable if its representations are sufficient for a task similar to the one it has been trained for. For example, in our context, we expect that a model trained on CIFAR-100 should transfer well to other datasets such as CIFAR-10. More formally:

Definition 21. Let $r_{\mathcal{T}_1}(\cdot)$ be the representations in output to the feature extractor obtained by training on task $\mathcal{T}_1$. In the same manner, let $r_{\mathcal{T}_2}$ be the representations obtained by training on $\mathcal{T}_2$. Let $(A_{\mathcal{T}_1 \to \mathcal{T}_2}, b_{\mathcal{T}_1 \to \mathcal{T}_2})$ be the parameters of the linear classifier trained to fit task $\mathcal{T}_2$ using the representation $r_{\mathcal{T}_1}$. We say that the model $\theta_{\mathcal{T}_1}$ is *α-transferable* to the task $\mathcal{T}_2$ when:

$$\mathcal{L}_{\mathcal{T}_1 \to \mathcal{T}_2} = \frac{1}{\alpha} \mathcal{L}_{\mathcal{T}_2} \tag{10.21}$$

where $\mathcal{L}_{\mathcal{T}}$ is the usual empirical loss on the validation set of task $\mathcal{T}$.

From the definition it is clear the α is well-defined only in the interval $[0, 1]$. When $\alpha = 1$, $\theta_{\mathcal{T}_1}$ is completely transferable for the task $\mathcal{T}_2$. The lower α, the less the model $\theta_{\mathcal{T}_1}$ can be used for $\mathcal{T}_2$. We would like to note that α depends on the distance d between the two tasks. When $\mathcal{T}_1 \equiv \mathcal{T}_2$, the model is 1-transferable by construction, while when $d(\mathcal{T}_1, \mathcal{T}_2)$ is high, we expect α to be very low.

10.7 Invertibility: Why ResNet work so well?

Residual networks and their several variants, are probably the most popular architecture for image recognition. The building block is of the form:

$$\begin{cases} x_{t+1} = x_t + \xi_{\theta_t}(x) & \text{for } dim(x_t) = dim(x_{t+1}) \\ x_{t+1} = W_t x_t + \xi_{\theta_t}(x) & \text{for } dim(x_t) \neq dim(x_{t+1}) \end{cases} \tag{10.22}$$

where $\xi_{\theta_t}(x)$ is the residual part to be learnt and W_t is a simple linear projection to match the dimensions. The output of the network is the result of multiple compositions of the residual map (10.22):

$$y(x) = x_T \circ x_{T-1} \circ \cdots \circ x_0(x) \tag{10.23}$$

Now, let us consider a ResNet architecture as a generation network T that maps a noise μ to a target distribution ν. In order to ease the argumentation, let us assume that μ and ν are equi-dimensional. By the Brenier's theorem Theorem 4 we know that, under squared L2 cost $c(x,y) = \|x-y\|_2^2$, the optimal transport map from a source distribution μ to a target ν is given by a gradient of a convex potential such that:

$$T(x) = x + \nabla\varphi(x) = \nabla\left[\frac{1}{2}\|x\|^2 + \varphi(x)\right] := \nabla u(x)$$

The requirement that φ is a convex function is typically very hard to enforce. However, we remark that $\xi_{\theta_t} = \nabla\varphi_t$ does not need to be convex. Moreover, for close distributions, convexity of $u(x)$ can be approximately satisfied, as $u(x)$ is very close to the (strictly convex) potential $\frac{1}{2}\|x\|^2$ corresponding to the identity map. Thus, if we consider the transport as a composition of many local optimal transports

$$\nabla u = \nabla u_T \circ \nabla u_{T-1} \circ \cdots \circ \nabla u_0$$

we can guarantee a nearly optimal solution to the transportation problem. Increasing the number of layers, the maps are more localized and smooth, satisfying more the convexity constraint of the local u_τ. We remark that in general the composition of gradients (of convex maps) is not a gradient of a convex function.

However, let us consider the intermediate densities $\rho_0 = x, \ldots, \rho_T = y$. If ρ_t and ρ_{t+1} are close enough, meaning that ρ_{t+1} is obtained by a small perturbation of the strictly convex potential $\frac{1}{2}\|x\|^2$ by ϕ, Proposition 15 says that $\nabla\phi$ gives the optimal transport for the squared quadratic cost. The forward pass computes the current compositional transport and the backward pass back-propagates the error to $\rho_T \to \rho_{T-1} \to \cdots \to \rho_1$. Thus, as long as we can guarantee that ρ_t and ρ_{t+1} are close enough, the residual network T_θ has, at convergence, $\rho_T = \nu$ with T_θ optimal in the Brenier's sense. Interestingly, if we consider the layer τ as 'time', the intermediate layers are the displacement interpolation defined in Def. 5.
From this argument, it is clear how the depth of the network is crucial to obtain smooth results. When T is not big enough, u is not locally convex and thus there are no guarantees on the transport. Instead, When $T \to +\infty$, local transport maps are infinitesimal perturbations of the identity function, whose potential is strictly convex.

Chapter 11

Conclusions

Deep Learning had and will have a tremendous impact both on the Machine Learning field and on the realm of real-world applications that involve our lives. However, together with the positive impacts, several issues still have to be solved, both from theoretical and practical perspectives.

Amongst all the issues discussed in this thesis, we addressed the following ones:

- **Robustness and accuracy trade-off**. With trivial tasks, very accurate models are not robust. However, not accurate and robust models are useless and, on the contrary, accurate but unreliable models cannot be employed where guarantees on their behavior are required. We provided an algorithm that allows to semantically trade-off between robustness and accuracy in an efficient way: we enforce robustness only where is strictly required allowing to preserve more accuracy;
- **Potentialities of a classifier**. In the DL literature, there are more and more sophisticated methodologies but it is not clear whether they work better than carefully tuned simpler established approaches. Rather than proposing new sophisticated framework we provided a connection between GANs, Optimal Transport and Adversarial Training, which allowed to show that a properly-trained classifier can solve other tasks other than classification;
- **Minimality**. Based on the above results, we introduced a new simple framework to obtain minimal representations. Our definition of minimality is weaker than its counterpart based in information theory. It does not require a stochastic description of neural networks or probability distributions;
- **Interpretability**. State-of-the-art DL models are poorly interpretable making them not well-suited for applications where human is involved. In view of this, we establish a set of desiderata for building interpretable models.

11.1 Future works

Some of the observations we have made throughout this thesis, pointed us to new ideas for future research. We summarize them in the following:

- **Robust GAN**: f-divergences like JSD or KL are too strong resulting in undesirable behaviors such as mode collapse and instability of the gradients. However, as we have seen, AT helps to soften the divergences making gradients more meaningful. Thus, one can substitute than standard GAN-discriminator with a robust version. The main limitation of this approach regards the computational complexity. Thus, an accurate investigation is required to analyze the trade-off between performance and time complexity.

- **Annealing of ϵ**. One of the issues of AT, is that it uses an isotropic ball. Where two or more manifolds are very close to each other, this can dramatically destroy the peculiarities of a single class. A direction to investigate is to anneal ϵ, such that $\epsilon \to 0$, as the training reaches the maximum epoch. Interestingly, this procedure is very similar to the training procedure introduced by Burgess et al., 2018 in the context of Information Bottleneck and VAEs.

- **Extensions to semi-supervised and unsupervised training**. One of the issues of supervised approaches is that they requires labelled datasets and in many contexts this is a great impediment. This motivates the extension to a semi-supervised setting where sources from similar domains are exploited. Moreover, most of the supervised dataset have been coarsely categorized. As a consequence, a classifier may discard important features. This motivates the unsupervised setting.

Appendix A

Miscellaneous

A.1 Danskin's theorem

Let us assume that ℓ is continuosly differentiable with respect to θ, then we have the following theorem:

Theorem 22 (Danskin's theorem). *Let $\mathcal{M}$ be a non-empty compact topological space and $g : \mathbb{R}^n \times S \to \mathbb{R}$ such that $g(\cdot, \delta)$ is differentiable with respect to any $\delta \in \mathcal{M}$ and $\nabla_\theta g(\theta, \delta)$ is continuous. Let us define $\delta^\star(\theta) = \arg\max_{\delta \in \mathcal{M}} g(\theta, \delta)$. Then the function:*

$$\psi(\theta) = \max_{\delta \in \mathcal{M}} g(\theta, \delta)$$

is locally Lipschitz continuous, and its directional derivative in the direction of y is:

$$\dot{\psi}(\theta, y) = \max_{\delta \in \delta^\star(\theta)} y^\top \nabla_\theta g(\theta, \delta)$$

Moreover, if for $\theta \in \mathbb{R}^n$ the set $\delta^\star(\theta)$ is composed by only one element, it holds:

$$\nabla \psi(\theta) = \nabla_\theta(\theta, \delta^\star(\theta))$$

A.2 Proof of implicit regularization

Proposition 23 (Implicit regularization). *Let $\phi(x)$ defined as in (10.8) and assume that its optimazion problem is strongly-concave. This can be guaranteed for example when $\nabla_x \varphi(x)$ is L-Lipschitz and $\lambda > L$, for some L. Then, the problem (10.9) is equivalent to minimize the regularized loss:*

$$\mathcal{L}(x; \theta) = \varphi(x) + \frac{1}{2} \|\nabla_x \varphi^\top(x) \nabla_x \varphi(x)\|_{\bar{F}^{-1}} + O\left(\frac{1}{\lambda^2}\right) \tag{A.1}$$

where $\bar{F} = F - \lambda I$ and F is the hessian of the negative log-likelihood $F(x) = \nabla_x^2 \log \widehat{y}(x)$.

Proof. If the problem is strongly concave we can stop to the second order expansion. In fact we can write:

$$\varphi(x') - \lambda c(x', x) \approx \varphi(x) + \nabla_x^\top \delta + \frac{1}{2} \delta^\top H \delta - \lambda \|\delta\|_2^2$$

and its maximization is given by:

$$\delta^\star = -(H - \lambda I)^{-1} \nabla_x \varphi(x)$$

Let us define $\bar{H} = H - \lambda I$. Plugging it into the expression for ϕ, we obtain:

$$\phi(x^\star) = \varphi(x) + \frac{1}{2}\nabla\varphi(x)^\top \bar{H}^{-1}\bar{H}\bar{H}^{-1}\nabla\varphi(x)$$

the reduces to:

$$\phi(x^\star) = \varphi(x) + \frac{1}{2}\nabla\varphi(x)^\top \bar{H}^{-1}\nabla\varphi(x) = \varphi(x) + \|\nabla\varphi(x)\|^2_{\bar{H}^{-1}}.$$

When the loss ℓ is of the form $-\log f(x)$ where $f(x)$ is a likelihood, then the hessian corresponds to the negative obserservation Fisher matrix F modulated by λI. ■

A.3 Datasets and Networks

Datasets. We use the following datasets:

(i) MNIST (LeCun et al., 1998): Classes consist of 10 digits from 0 to 9 and images have size 28 × 28 with 1 gray channel.

(ii) CIFAR-10 (Krizhevsky, 2009): it consists of 10 classes: *airplane, car, truck, ship, horse, dog, cat, bird, frog, deer*. Images are RGB of size 32 × 32. Each class contains 5000 images.

(iii) CIFAR-100 (Krizhevsky, 2009): this dataset is similar CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs). In this work we will make use only of fine labels.

(iv) TinyImageNet (*TinyImageNet*) is a tiny version of ImageNet with 200 classes containing 600 RGB images each of size 64 × 64.

(v) RestrictedImageNet introduced in Tsipras et al., 2019. It is a short version of ImageNet where semantically similar classes from ImageNet are group together into 9 super-classes shown in Table A.1. Images are RGB of cropped size 224 × 224.

TABLE A.1: Classes used in the RestrictedImageNet model. The class ranges are inclusive.

Class	Corresponding ImageNet Classes
"Dog"	151 to 268
"Cat"	281 to 285
"Frog"	30 to 32
"Turtle"	33 to 37
"Bird"	80 to 100
"Primate"	365 to 382
"Fish"	389 to 397
"Crab"	118 to 121
"Insect"	300 to 319

For all datasets, images are normalized to have pixel intensities between $[0, 1]$. For the CIFAR datasets, we use standard data-augmentation which involves mirror flipping with probability of 0.5 and random crops of size 32×32 after padding images by 4 pixels on each side.

Architectures. Our experiments we will use ResNets (He et al., 2016) and its wide version (Zagoruyko and Komodakis, 2016). In particular, we will adopt:

(i) **ResNet-18:** ResNet with 18 layers.

(ii) **ResNet-50:** ResNet with 50 layers.

(iii) **W-16-10:** Wide-Residual network of Zagoruyko and Komodakis, 2016 with 16 layers, a widening factor of 10.

(iv) **W-28-10:** Wide-Residual network with 28 layers, a widening factor of 10.

(v) **W-40-10:** Wide-Residual network with 50 layers, a widening factor of 10.

All networks are trained with stochastic gradient descent (SGD), Nesterov's momentum of 0.9, mini-batch size of 128, weight decay of 5×10^{-4} and no dropbox. The adversarial vulnerability of neural networks increases with the number of output classes Fawzi, Fawzi, and Fawzi, 2018. In this context, we would like to emphasize that the TibyImageNet dataset with 200 classes is a viable dataset for benchmarking adversarial learning algorithms; this dataset is however less popular in the literature which primarily focuses on MNIST and CIFAR-10.

Appendix B

Minimum Energy Models

B.1 Experimental Setup

B.1.1 Datasets

For our experiments, we use the RestrictedImageNet and CIFAR-10 datasets.

B.1.2 Models

We use the standard ResNet-18 architecture (He et al., 2016) for our adversarially trained classifiers on all datasets. Every model is trained with data augmentation, momentum of 0.9 and weight decay of 5×10^{-4}. Other hyperparameters are provided in Tables B.3 and B.2.

TABLE B.1: Standard hyperparameters for training the models.

Dataset	Epochs	LR	Batch Size	Drop LR
RestrictedImageNet	100	0.1	128	0.1 at epochs $\in [30, 60]$
CIFAR-10	200	0.1	128	0.2 at epochs $\in [60, 120, 160]$

B.1.3 Adversarial training and natural training

We employ the adversarial training methodology described in Sec. 8.2 and introduced by (Madry et al., 2017). We consider adversarial perturbations in ℓ_2 norm. Unless otherwise specified, we use the values of ϵ provided in Table B.2 to train/evaluate our models. Table B.2 provides the hyper-parameters for natural training.

TABLE B.2: Hyperparameters used for AT.

Dataset	ϵ	# steps	Step size
RestrictedImageNet	3.	7	0.5
CIFAR-10	1.	7	0.5

TABLE B.3: Hyperparameters used for NT.

Dataset	λ	**# steps**	**Step size**
RestrictedImageNet	1e5	7	0.1
CIFAR-10	-	7	0.1

B.1.4 Image Transportation

	Dataset	ϵ	**# steps**	**Step size**
Robust	RestrictedImageNet	30	50	0.5
	CIFAR-10	30	50	0.5
Natural	RestrictedImageNet	30	50	0.5
	CIFAR-10	30	50	0.5

B.1.5 Image Interpolations

	Dataset	ϵ	**# steps**	**Step size**
Robust	RestrictedImageNet	-	10	0.5
	CIFAR-10	-	10	0.5
Natural	RestrictedImageNet	-	10	0.5
	CIFAR-10	-	10	0.5

B.1.6 Image Generation

	Dataset	ϵ	**# steps**	**Step size**
Robust	RestrictedImageNet	30	25	0.5
	CIFAR-10	30	25	0.5
Natural	RestrictedImageNet	30	25	0.5
	CIFAR-10	30	25	0.5

Bibliography

Achille, Alessandro and Stefano Soatto (2018). “Emergence of invariance and disentanglement in deep representations”. In: *The Journal of Machine Learning Research* 19.1, pp. 1947–1980.

Agustsson, Eirikur et al. (2017). “Optimal transport maps for distribution preserving operations on latent spaces of generative models”. In: *arXiv preprint arXiv:1711.01970*.

Akima, Hiroshi (1970). “A new method of interpolation and smooth curve fitting based on local procedures”. In: *Journal of the ACM* 17.4, pp. 589–602.

Al-Naymat, Ghazi, Sanjay Chawla, and Javid Taheri (2009). “SparseDTW: a novel approach to speed up dynamic time warping”. In: *Proceedings of the Eighth Australasian Data Mining Conference-Volume 101*. Australian Computer Society, Inc., pp. 117–127.

Albinali, F. et al. (2010). “Using Wearable Activity Type Detection to Improve Physical Activity Energy Expenditure Estimation”. In: *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*. UbiComp ’10. Copenhagen, Denmark, pp. 311–320. ISBN: 978-1-60558-843-8.

Alvarez-Melis, David and Tommi S Jaakkola (2018). “Towards robust interpretability with self-explaining neural networks”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., pp. 7786–7795.

Ambrosio, Luigi, Nicola Gigli, and Giuseppe Savaré (2008). *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media.

Anguita, Davide et al. (2013). “A public domain dataset for human activity recognition using smartphones”. In: *21th Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 437–442.

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). “Wasserstein gan”. In: *arXiv preprint arXiv:1701.07875*.

Arsand, Eirik, Ragnhild Varmedal, and Gunnar Hartvigsen (2007). “Usability of a mobile self-help tool for people with diabetes: the easy health diary”. In: *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*. IEEE, pp. 863–868.

Aswolinskiy, Witali, René Felix Reinhart, and Jochen Steil (2016). “Time series classification in reservoir-and model-space: a comparison”. In: *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, pp. 197–208.

Athalye, Anish, Nicholas Carlini, and David Wagner (2018). “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples”. In: *arXiv:1802.00420*.

Bagherinezhad, Hessam et al. (2018). “Label Refinery: Improving ImageNet Classification through Label Progression”. In: *arXiv:1805.02641*.

Bailly, Adeline et al. (2015). “Bag-of-Temporal-SIFT-Words for Time Series Classification”. In: *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*.

— (2016). “Dense Bag-of-Temporal-SIFT-Words for Time Series Classification”. In:

Barazandeh, Babak et al. (2017). "Robust Sparse Representation-Based Classification Using Online Sensor Data for Monitoring Manual Material Handling Tasks". In: *IEEE Transactions on Automation Science and Engineering*.

Bastani, Kaveh et al. (2013). "Fault diagnosis using an enhanced relevance vector machine (RVM) for partially diagnosable multistation assembly processes". In: *IEEE Transactions on Automation Science and Engineering* 10.1, pp. 124–136.

Batista, Gustavo EAPA, Xiaoyue Wang, and Eamonn J Keogh (2011). "A Complexity-Invariant Distance Measure for Time Series." In: *SDM*. Vol. 11. SIAM, pp. 699–710.

Baydogan, Mustafa Gokce and George Runger (2015). "Learning a symbolic representation for multivariate time series classification". In: *Data Mining and Knowledge Discovery* 29.2, pp. 400–422.

Baydogan, Mustafa Gokce, George Runger, and Eugene Tuv (2013). "A bag-of-features framework to classify time series". In: *IEEE transactions on pattern analysis and machine intelligence* 35.11, pp. 2796–2802.

Beghi, Alessandro et al. (2014). "A one-class svm based tool for machine learning novelty detection in hvac chiller systems". In: *IFAC Proceedings Volumes* 47.3, pp. 1953–1958.

Belgioioso, G. et al. (2014). "A machine learning based approach for gesture recognition from inertial measurements". In: *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, pp. 4899–4904.

Belkin, Mikhail, Partha Niyogi, and Vikas Sindhwani (2006). "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples". In: *Journal of machine learning research* 7.Nov, pp. 2399–2434.

Benamou, Jean-David and Yann Brenier (2000). "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem". In: *Numerische Mathematik* 84.3, pp. 375–393.

Bentley, Jon Louis (1975). "Multidimensional binary search trees used for associative searching". In: *Communications of the ACM* 18.9, pp. 509–517.

Bergman, Richard N (1989). "Toward physiological understanding of glucose tolerance: minimal-model approach". In: *Diabetes* 38.12, pp. 1512–1527.

Berndt, Donald J and James Clifford (1994). "Using Dynamic Time Warping to Find Patterns in Time Series." In: *KDD workshop*. Vol. 10. 16. Seattle, WA, pp. 359–370.

Biau, GÃŠrard (2012). "Analysis of a random forests model". In: *Journal of Machine Learning Research* 13.Apr, pp. 1063–1095.

Bishop, C.M. et al. (2006). *Pattern recognition and machine learning*. Vol. 4. 4. Springer New York.

Bissacco, Alessandro, Alessandro Chiuso, and Stefano Soatto (2007). "Classification and recognition of dynamical models: The role of phase, independent components, kernels and optimal transport". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.11, pp. 1958–1972.

Blanchet, Jose and Karthyek Murthy (2019). "Quantifying distributional model risk via optimal transport". In: *Mathematics of Operations Research* 44.2, pp. 565–600.

Bon, Arianne C van, Dorien Dragt, and J Hans DeVries (2012). "Significant time until catheter occlusion alerts in currently marketed insulin pumps at two basal rates". In: *Diabetes technology & therapeutics* 14.5, pp. 447–448.

Bon, Arianne C van et al. (2011). "Insulin glulisine compared to insulin aspart and to insulin lispro administered by continuous subcutaneous insulin infusion in patients with type 1 diabetes: a randomized controlled trial". In: *Diabetes technology & therapeutics* 13.6, pp. 607–614.

Boser, Bernhard E, Isabelle M Guyon, and Vladimir N Vapnik (1992). "A training algorithm for optimal margin classifiers". In: *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, pp. 144–152.

Boyd, Stephen and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Breiman, Leo et al. (1984). *Classification and regression trees*. CRC press.

Brenier, Yann (1991). "Polar factorization and monotone rearrangement of vector-valued functions". In: *Communications on pure and applied mathematics* 44.4, pp. 375–417.

Breunig, Markus M et al. (2000). "LOF: identifying density-based local outliers". In: *ACM sigmod record*. Vol. 29. 2. ACM, pp. 93–104.

Brock, Andrew, Jeff Donahue, and Karen Simonyan (2018). "Large scale gan training for high fidelity natural image synthesis". In: *arXiv preprint arXiv:1809.11096*.

Broderick, John A, Lindsay V Allen, and Dawn M Tilbury (2011). "Anomaly detection without a pre-existing formal model: Application to an industrial manufacturing system". In: *Automation Science and Engineering (CASE), 2011 IEEE Conference on*. IEEE, pp. 169–174.

Burgess, Christopher P et al. (2018). "Understanding disentangling in

$$\backslash$$

beta $-VAE$". In: *arXiv preprint arXiv:1804.03599*.

Candan, K Selçuk et al. (2012). "sDTW: computing DTW distances using locally relevant constraints based on salient feature alignments". In: *Proceedings of the VLDB Endowment* 5.11, pp. 1519–1530.

Carlini, Nicholas and David Wagner (2017). "Adversarial examples are not easily detected: Bypassing ten detection methods". In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, pp. 3–14.

Cayton, Lawrence (2005). "Algorithms for manifold learning". In: *Univ. of California at San Diego Tech. Rep*, pp. 1–17.

Cenedese, Angelo, Gian Antonio Susto, and Matteo Terzi (2016). "A Parsimonious Approach for Activity Recognition with Wearable Devices: an Application to Cross-country Skiing". In: *European Control Conference*, pp. 1243–1252.

Cenedese, Angelo et al. (2015). "Home Automation Oriented Gesture Classification From Inertial Measurements". In: *IEEE Trans. on Automation Science and Engineering* 12.4, pp. 1200–1210.

Cescon, Marzia et al. (2016). "Early detection of infusion set failure during insulin pump therapy in type 1 diabetes". In: *Journal of diabetes science and technology* 10.6, pp. 1268–1276.

Chen, Huanhuan et al. (2015a). "Model Metric Co-Learning for Time Series Classification." In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. AAAI Press, pp. 3387–3394.

Chen, Lei and Raymond Ng (2004). "On the marriage of lp-norms and edit distance". In: *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, pp. 792–803.

Chen, Lei, M Tamer Özsu, and Vincent Oria (2005). "Robust and fast similarity search for moving object trajectories". In: *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. ACM, pp. 491–502.

Chen, Wei-Yu et al. (2019). "A Closer Look at Few-shot Classification". In: *CoRR* abs/1904.04232. arXiv: 1904.04232. URL: http://arxiv.org/abs/1904.04232.

Chen, Zhihua et al. (2015b). “Kernel sparse representation for time series classification”. In: *Information Sciences* 292, pp. 15–26.

Cheng, Zhijin et al. (2016). “Case studies of fault diagnosis and energy saving in buildings using data mining techniques”. In: *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*. IEEE, pp. 646–651.

Clifton, Lei et al. (2013). “Gaussian processes for personalized e-health monitoring with wearable sensors”. In: *IEEE Trans. on Biomedical Eng.* 60.1, pp. 193–197.

Cobelli, Claudio et al. (2009). “Diabetes: models, signals, and control”. In: *IEEE reviews in biomedical engineering* 2, p. 54.

Control, Diabetes and Complications Trial Research Group (1993). “The effect of intensive treatment of diabetes on the development and progression of long-term complications in insulin-dependent diabetes mellitus”. In: *New England journal of medicine* 329.14, pp. 977–986.

Cortes, Corinna and Vladimir Vapnik (1995). “Support-vector networks”. In: *Machine learning* 20.3, pp. 273–297.

Cox, David R (1958). “The regression analysis of binary sequences”. In: *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 215–242.

Csurka, Gabriella et al. (2004). “Visual categorization with bags of keypoints”. In: *Workshop on statistical learning in computer vision, ECCV*. Vol. 1. 1-22. Prague, pp. 1–2.

Cuturi, Marco (2011). “Fast global alignment kernels”. In: *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 929–936.

— (2013). “Sinkhorn distances: Lightspeed computation of optimal transport”. In: *Advances in neural information processing systems*, pp. 2292–2300.

Cuturi, Marco and Arnaud Doucet (2011). “Autoregressive kernels for time series”. In: *arXiv preprint arXiv:1101.0673*.

Cuturi, Marco et al. (2007). “A kernel for time series based on global alignments”. In: *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*. Vol. 2. IEEE, pp. II–413.

Darlow, Luke N et al. (2018). “CINIC-10 is not ImageNet or CIFAR-10”. In: *arXiv preprint arXiv:1810.03505*.

Del Favero, Simone et al. (2014). “Real-time detection of Glucose Sensor and Insulin Pump Faults in an Artificial Pancreas.” In: *IFAC Proceedings Volumes* 47.3, pp. 1941–1946.

Deng, Houtao et al. (2013). “A time series forest for classification and feature extraction”. In: *Information Sciences* 239, pp. 142–153.

Deng, Jia et al. (2009). “Imagenet: A large-scale hierarchical image database”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, pp. 248–255.

Deng, Jia et al. (2010). “What Does Classifying More Than 10,000 Image Categories Tell Us?” In: *Lecture Notes in Computer Science*, pp. 71–84. ISSN: 1611-3349. DOI: 10.1007/978-3-642-15555-0_6. URL: http://dx.doi.org/10.1007/978-3-642-15555-0_6.

Deng, Jia et al. (2014). “Large-Scale Object Classification Using Label Relation Graphs”. In: *Lecture Notes in Computer Science*, pp. 48–64. ISSN: 1611-3349. DOI: 10.1007/978-3-319-10590-1_4. URL: http://dx.doi.org/10.1007/978-3-319-10590-1_4.

Dhillon, Guneet S. et al. (2019). *A Baseline for Few-Shot Image Classification*. arXiv: 1909.02729 [cs.LG].

Dietterich, T.G. (1998). “Approximate statistical tests for comparing supervised classification learning algorithms”. In: *Neural Computation* 10.7, pp. 1895–1923.

Ding, Hui et al. (2008). "Querying and mining of time series data: experimental comparison of representations and distance measures". In: *VLDB Endowment* 1.2, pp. 1542–1552.

Ding, Zhiguo and Minrui Fei (2013). "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window". In: *IFAC Proceedings Volumes* 46.20, pp. 12–17.

Ellingsen, Christian et al. (2009). "Safety constraints in an artificial pancreatic β cell: an implementation of model predictive control with insulin on board". In: *Journal of diabetes science and technology* 3.3, pp. 536–544.

Engstrom, Logan et al. (2019). "Learning Perceptually-Aligned Representations via Adversarial Robustness". In: *ArXiv preprint arXiv:1906.00945*.

Eruhimov, Victor, Vladimir Martyanov, and Eugene Tuv (2007). "Constructing high dimensional feature space for time series classification". In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, pp. 414–421.

Esteva, Andre et al. (2019). "A guide to deep learning in healthcare". In: *Nature medicine* 25.1, p. 24.

Facchinetti, Andrea et al. (2013). "An online failure detection method of the glucose sensor-insulin pump system: improved overnight safety of type-1 diabetic subjects". In: *IEEE Transactions on Biomedical Engineering* 60.2, pp. 406–416.

Facchinetti, Andrea et al. (2014). "Modeling the glucose sensor error". In: *IEEE Transactions on Biomedical Engineering* 61.3, pp. 620–629.

Fasel, B. et al. (2015). "An inertial sensor-based system for spatio-temporal analysis in classic cross-country skiing diagonal technique". In: *Journal of Biomechanics*, pp. –. ISSN: 0021-9290. DOI: `http://dx.doi.org/10.1016/j.jbiomech.2015.07.001`. URL: `http://www.sciencedirect.com/science/article/pii/S0021929015003784`.

Fawzi, Alhussein, Hamza Fawzi, and Omar Fawzi (2018). "Adversarial vulnerability for any classifier". In: *arXiv:1802.08686*.

Fawzi, Alhussein, Omar Fawzi, and Pascal Frossard (2018). "Analysis of classifiers robustness to adversarial perturbations". In: *Machine Learning* 107.3, pp. 481–508.

Figalli, Alessio, Young-Heon Kim, and Robert J McCann (2011). "When is multidimensional screening a convex program?" In: *Journal of Economic Theory* 146.2, pp. 454–478.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2001). *The elements of statistical learning*. Vol. 1. Springer series in statistics Springer, Berlin.

Frogner, Charlie et al. (2015). "Learning with a Wasserstein loss". In: *Advances in Neural Information Processing Systems*, pp. 2053–2061.

Fu, Tak-chung (2011). "A review on time series data mining". In: *Engineering Applications of Artificial Intelligence* 24.1, pp. 164–181.

G.A., Susto et al. (2016). "Supervised Aggregative Feature Extraction for Big Data Time Series Regression". In: *IEEE Transactions on Industrial Informatics*.

Garrett, Deon et al. (2003). "Comparison of linear, nonlinear, and feature selection methods for EEG signal classification". In: *IEEE Transactions on neural systems and rehabilitation engineering* 11.2, pp. 141–144.

Geurts, Pierre (2001). "Pattern extraction for time series classification". In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, pp. 115–127.

Girshick, Ross et al. (2013). "Rich feature hierarchies for accurate object detection and semantic segmentation". In: eprint: `1311.2524`. URL: `https://arxiv.org/pdf/1311.2524`.

Goldstein, Markus (2012). “FastLOF: An expectation-maximization based local outlier detection algorithm”. In: *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, pp. 2282–2285.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016a). “Deep learning”. In: *2015*.

— (2016b). *Deep Learning*. MIT Press.

Goodfellow, Ian et al. (2014). “Generative adversarial nets”. In: *NIPS*, pp. 2672–2680.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy (2014). “Explaining and harnessing adversarial examples”. In: *arXiv:1412.6572*.

Gowing, Marc et al. (2014). “Kinect vs. Low-cost Inertial Sensing For Gesture Recognition”. In: *Lecture Notes in Computer Science* 8325, pp. 484–495.

Grabocka, Josif et al. (2014). “Learning time-series shapelets”. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 392–401.

Groves, Paul D (2013). *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech house.

Guilhem, I. et al. (2009). “Insulin pump failures are still frequent: a prospective study over 6 years from 2001 to 2007”. In: *Diabetologia* 52, pp. 2662–2664.

Guo, Chuan et al. (2017). “On calibration of modern neural networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1321–1330.

Hamlet, Connor et al. (2017). “An incremental and approximate local outlier probability algorithm for intrusion detection and its evaluation”. In: *Journal of Cyber Security Technology* 1.2, pp. 75–87.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning*. Springer-Verlag New York.

He, Kaiming et al. (2016). “Identity mappings in deep residual networks”. In: *arXiv:1603.05027*.

He, Zhen-Yu and Lian-Wen Jin (2008). “Activity recognition from acceleration data using AR model representation and SVM”. In: *2008 International Conference on Machine Learning and Cybernetics*. Vol. 4. IEEE, pp. 2245–2250.

Herrero, Pau et al. (2012). “Robust fault detection system for insulin pump therapy using continuous glucose monitoring”. In: *Journal of diabetes science and technology* 6.5, pp. 1131–1141.

Hills, Jon et al. (2014). “Classification of time series by shapelet transformation”. In: *Data Mining and Knowledge Discovery* 28.4, pp. 851–881.

Holst, A. and A. Jonasson (2013). “Classification of movement patterns in skiing.” In: *SCAI*, pp. 115–124.

Hotelling, Harold (1933). “Analysis of a complex of statistical variables into principal components.” In: *Journal of educational psychology* 24.6, p. 417.

Hovorka, Roman et al. (2004). “Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes”. In: *Physiological measurement* 25.4, p. 905.

Howsmon, Daniel P et al. (2017). “Continuous glucose monitoring enables the detection of Losses in Infusion Set Actuation (LISAs)”. In: *Sensors* 17.1, p. 161.

Huang, Norden E et al. (1998b). “The Empirical Mode Decomposition and the Hilbert Spectrum for Nonlinear and Non-stationary Time Series Analysis”. In: *Royal Society of London A: Mathematical, Physical and Engineering Sciences*. 1971, pp. 903–995.

— (1998a). “The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis”. In: *Proceedings of the Royal Society of*

London A: Mathematical, Physical and Engineering Sciences. Vol. 454. 1971. The Royal Society, pp. 903–995.

Hyvärinen, Aapo, Juha Karhunen, and Erkki Oja (2004). *Independent component analysis*. Vol. 46. John Wiley & Sons.

Ibitoye, Olakunle, Omair Shafiq, and Ashraf Matrawy (2019). "Analyzing Adversarial Attacks Against Deep Learning for Intrusion Detection in IoT Networks". In: *arXiv preprint arXiv:1905.05137*.

Ioffe, S. and C. Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *arXiv:1502.03167*.

Islam, SM Riazul et al. (2015). "The internet of things for health care: a comprehensive survey". In: *IEEE Access* 3, pp. 678–708.

Jaeger, Herbert (2001). "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note". In: *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* 148, p. 34.

— (2002). "Adaptive nonlinear system identification with echo state networks". In: *Advances in neural information processing systems*, pp. 593–600.

Jahankhani, Pari, Vassilis Kodogiannis, and Kenneth Revett (2006). "EEG signal classification using wavelet feature extraction and neural networks". In: *IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing (JVA'06)*. IEEE, pp. 120–124.

Jensen, Willis A et al. (2006). "Effects of parameter estimation on control chart properties: a literature review". In: *Journal of Quality Technology* 38.4, p. 349.

Jeong, Young-Seon, Myong K Jeong, and Olufemi A Omitaomu (2011). "Weighted dynamic time warping for time series classification". In: *Pattern Recognition* 44.9, pp. 2231–2240.

Jiang, Zhuolin, Zhe Lin, and Larry S Davis (2011). "Learning a discriminative dictionary for sparse coding via label consistent K-SVD". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, pp. 1697–1704.

— (2013). "Label consistent K-SVD: Learning a discriminative dictionary for recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.11, pp. 2651–2664.

Jolliffe, Ian (2002). *Principal component analysis*. Wiley Online Library.

Kadous, Mohammed Waleed (1999). "Learning Comprehensible Descriptions of Multivariate Time Series." In: *ICML*, pp. 454–463.

Kanderian, Sami S et al. (2009). *Identification of intraday metabolic profiles during closed-loop glucose control in individuals with type 1 diabetes*.

Keogh, Eamonn and Chotirat Ann Ratanamahatana (2005). "Exact indexing of dynamic time warping". In: *Knowledge and information systems* 7.3, pp. 358–386.

Keogh, Eamonn et al. (2001). "Dimensionality reduction for fast similarity search in large time series databases". In: *Knowledge and information Systems* 3.3, pp. 263–286.

Keogh, Eamonn J and Michael J Pazzani (2001). "Derivative Dynamic Time Warping." In: *Sdm*. Vol. 1. SIAM, pp. 5–7.

Kingma, Diederik P and Max Welling (2013). "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114*.

Kolter, J Zico and Eric Wong (2017). "Provable defenses against adversarial examples via the convex outer adversarial polytope". In: *arXiv preprint arXiv:1711.00851*.

Kovács, L et al. (2006). "LPV fault detection of glucose-insulin system". In: *Control and Automation, 2006. MED'06. 14th Mediterranean Conference on*. IEEE, pp. 1–5.

Kriegel, Hans-Peter, Arthur Zimek, et al. (2008). "Angle-based outlier detection in high-dimensional data". In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 444–452.

Krizhevsky, A. (2009). "Learning multiple layers of features from tiny images". MA thesis. Computer Science, University of Toronto.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *NIPS*.

Kropff, Jort et al. (2015). "2 month evening and night closed-loop glucose control in patients with type 1 diabetes under free-living conditions: a randomised crossover trial". In: *The lancet Diabetes & endocrinology* 3.12, pp. 939–947.

Kruskal, Joseph B and Myron Wish (1978). *Multidimensional scaling*. Vol. 11. Sage.

Kudo, Mineichi, Jun Toyama, and Masaru Shimbo (1999). "Multidimensional curve classification using passing-through regions". In: *Pattern Recognition Letters* 20.11, pp. 1103–1111.

LeCun, Y. et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Ledig, Christian et al. (2017). "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690.

Leslie, Christina and Rui Kuang (2004). "Fast string kernels using inexact matching for protein sequences". In: *Journal of Machine Learning Research* 5.Nov, pp. 1435–1455.

Li, Hao et al. (2018). "Visualizing the loss landscape of neural nets". In: *Advances in Neural Information Processing Systems*, pp. 6389–6399.

Li, Yang, Junyuan Hong, and Huanhuan Chen (2016). "Sequential Data Classification in the Space of Liquid State Machines". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 313–328.

Lin, Jessica, Rohan Khade, and Yuan Li (2012). "Rotation-invariant similarity in time series using bag-of-patterns representation". In: *Journal of Intelligent Information Systems* 39.2, pp. 287–315.

Lin, Jessica et al. (2003a). "A symbolic representation of time series, with implications for streaming algorithms". In: *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*. ACM, pp. 2–11.

— (2003b). "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms". In: *ACM SIGMOD Research Issues in Data Mining and Knowledge Discovery*, pp. 2–11.

Lin, Jessica et al. (2007). "Experiencing SAX: a novel symbolic representation of time series". In: *Data Mining and knowledge discovery* 15.2, pp. 107–144.

Lines, Jason and Anthony Bagnall (2015). "Time series classification with ensembles of elastic distance measures". In: *Data Mining and Knowledge Discovery* 29.3, pp. 565–592.

Lines, Jason et al. (2012). "A shapelet transform for time series classification". In: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 289–297.

Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou (2008). "Isolation forest". In: *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, pp. 413–422.

Liu, J., Z. Pan, and X. Li (2010). "An Accelerometer-Based Gesture Recognition Algorithm and its Application for 3D Interaction". In: *Computer Science and Information Systems* 7.1.

Ljung, Lennart (1998). "System identification". In: *Signal Analysis and Prediction*. Springer, pp. 163–173.

Lowe, David G (1999). "Object recognition from local scale-invariant features". In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee, pp. 1150–1157.

Lu, Yiping et al. (2017). "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations". In: *arXiv preprint arXiv:1710.10121*.

Luc, Pauline et al. (2016). "Semantic segmentation using adversarial networks". In: *arXiv preprint arXiv:1611.08408*.

Luhn, Hans Peter (1957). "A statistical approach to mechanized encoding and searching of literary information". In: *IBM Journal of research and development* 1.4, pp. 309–317.

Ma, Qianli et al. (2016). "Functional echo state network for time series classification". In: *Information Sciences* 373, pp. 1–20.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov, pp. 2579–2605.

Madgwick, S. (2010). "An efficient orientation filter for inertial and inertial/magnetic sensor arrays". In:

Madry, Aleksander et al. (2017). "Towards deep learning models resistant to adversarial attacks". In: *arXiv:1706.06083*.

Mahmoudi, Zeinab et al. (2016). "Comparison of three nonlinear filters for fault detection in continuous glucose monitors". In: *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*. IEEE, pp. 3507–3510.

Mairal, Julien, Francis Bach, and Jean Ponce (2012). "Task-driven dictionary learning". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.4, pp. 791–804.

Mairal, Julien et al. (2008). "Discriminative learned dictionaries for local image analysis". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, pp. 1–8.

Mairal, Julien et al. (2009). "Supervised dictionary learning". In: *Advances in neural information processing systems*, pp. 1033–1040.

Man, Chiara Dalla et al. (2014). "The UVA/PADOVA type 1 diabetes simulator: new features". In: *Journal of diabetes science and technology* 8.1, pp. 26–34.

Manganaris, Stefanos (1997). *Supervised classification with temporal data*. Vanderbilt University.

Marteau, Pierre-François (2009). "Time warp edit distance with stiffness adjustment for time series matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.2, pp. 306–318.

Mathie, M.J. et al. (2004). "Accelerometry: providing an integrated, practical method for long-term, ambulatory monitoring of human movement". In: *Physiological Measurement* 25.2, R1. URL: http://stacks.iop.org/0967-3334/25/i=2/a=R01.

McAdams, Brooke H and Ali A Rizvi (2016). "An Overview of Insulin Pumps and Glucose Sensors for the Generalist". In: *Journal of clinical medicine* 5.1, p. 5.

McCann, Robert J (1997). "A convexity principle for interacting gases". In: *Advances in mathematics* 128.1, pp. 153–179.

McLachlan, Geoffrey (2004). *Discriminant analysis and statistical pattern recognition*. Vol. 544. John Wiley & Sons.

Medtronic Products Website: The MiniMed 670G Insulin Pump System. https://www.medtronicdiabetes.com/products/minimed-670g-insulin-pump-system. Accessed: 2017-09-18.

Mika, Sebastian et al. (1998). "Kernel PCA and De-Noising in Feature Spaces." In: *NIPS*. Vol. 11, pp. 536–542.

Miller, George A (1995). "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11, pp. 39–41.

Monge, Gaspard (1781). "Mémoire sur la théorie des déblais et des remblais". In: *Histoire de l'Académie Royale des Sciences de Paris*.

Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard (2016). "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582.

Moosavi-Dezfooli, Seyed-Mohsen et al. (2017). "Universal adversarial perturbations". In: *arXiv:1610.08401*.

Moosavi-Dezfooli, Seyed-Mohsen et al. (2018). "Robustness via curvature regularization, and vice versa". In: eprint: 1811.09716. URL: https://arxiv.org/pdf/1811.09716.

Morris, Dan et al. (2014). "RecoFit: Using a Wearable Sensor to Find, Recognize, and Count Repetitive Exercises". In: *SIGCHI Human Factors in Computing Systems*, pp. 3225–3234.

Mueen, Abdullah, Eamonn Keogh, and Neal Young (2011). "Logical-shapelets: an expressive primitive for time series classification". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 1154–1162.

Müller, Meinard (2007). "Dynamic time warping". In: *Information retrieval for music and motion*, pp. 69–84.

Nair, Vinod and Geoffrey E Hinton (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Narayanan, Hariharan and Sanjoy Mitter (2010). "Sample complexity of testing the manifold hypothesis". In: *Advances in Neural Information Processing Systems*, pp. 1786–1794.

Nassif, Ali Bou et al. (2019). "Speech recognition using deep neural networks: A systematic review". In: *IEEE Access* 7, pp. 19143–19165.

Netzer, Yuval et al. (2011). "Reading digits in natural images with unsupervised feature learning". In:

Nowozin, Sebastian, Botond Cseke, and Ryota Tomioka (2016). "f-gan: Training generative neural samplers using variational divergence minimization". In: *Advances in neural information processing systems*, pp. 271–279.

Oviedo, Silvia et al. (2017). "A review of personalized blood glucose prediction strategies for T1DM patients". In: *International journal for numerical methods in biomedical engineering* 33.6, e2833.

Papernot, Nicolas, Patrick McDaniel, and Ian Goodfellow (2016). "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples". In: *arXiv:1605.07277*.

Papernot, Nicolas et al. (2016). "Practical black-box attacks against deep learning systems using adversarial examples". In: *arXiv:1602.02697*.

Pathak, Deepak et al. (2016). "Context encoders: Feature learning by inpainting". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544.

Petitjean, François, Alain Ketterlin, and Pierre Gançarski (2011). "A global averaging method for dynamic time warping, with applications to clustering". In: *Pattern Recognition* 44.3, pp. 678–693.

Petitjean, François et al. (2016). "Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm". In: *Knowledge and Information Systems* 47.1, pp. 1–26.

Peyré, Gabriel and Marco Cuturi (2018). "Computational Optimal Transport". In: *arXiv:1803.00567*.

Pokrajac, Dragoljub, Aleksandar Lazarevic, and Longin Jan Latecki (2007). "Incremental local outlier detection for data streams". In: *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*. IEEE, pp. 504–515.

Qayyum, Adnan et al. (2019). "Securing Connected & Autonomous Vehicles: Challenges Posed by Adversarial Machine Learning and The Way Forward". In: *arXiv preprint arXiv:1905.12762*.

Rabiner, Lawrence and Biing-Hwang Juang (1993). "Fundamentals of speech recognition". In:

Radford, Alec, Luke Metz, and Soumith Chintala (2015). "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434*.

Raghunathan, Aditi et al. (2019). "Adversarial Training Can Hurt Generalization". In: *arXiv preprint arXiv:1906.06032*.

Rajagopalan, Venkatesh and Asok Ray (2006). "Symbolic time series analysis via wavelet-based partitioning". In: *Signal Processing* 86.11, pp. 3309–3320.

Rakthanmanon, Thanawin and Eamonn Keogh (2013). "Fast shapelets: A scalable algorithm for discovering time series shapelets". In: *Proceedings of the 13th SIAM international conference on data mining*. SIAM, pp. 668–676.

Ramkissoon, Charrise M et al. (2017). "A Review of Safety and Hazards Associated With the Artificial Pancreas". In: *IEEE reviews in biomedical engineering* 10, pp. 44–62.

Rasmussen, Carl Edward (2006). "Gaussian processes for machine learning". In:

Renard, Eric et al. (2016). "Day-and-night closed-loop glucose control in patients with type 1 diabetes under free-living conditions: results of a single-arm 1-month experience compared with a previously reported feasibility study of evening and night at home". In: *Diabetes Care* 39.7, pp. 1151–1160.

Roberts, Peter Sykacek Stephen (2002). "Bayesian time series classification". In: *Advances in Neural Information Processing Systems* 14, p. 937.

Rodríguez, Juan J and Carlos J Alonso (2004). "Interval and dynamic time warping-based decision trees". In: *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, pp. 548–552.

Rodríguez, Juan J, Carlos J Alonso, and Henrik Boström (2001). "Boosting interval based literals". In: *Intelligent Data Analysis* 5.3, pp. 245–262.

Rodríguez, Juan José, Carlos J Alonso, and José A Maestro (2005). "Support vector machines of interval-based features for time series classification". In: *Knowledge-Based Systems* 18.4, pp. 171–178.

Rojas, Ruben, Winston Garcia-Gabin, and B Wayne Bequette (2011). "Multivariate statistical analysis to detect insulin infusion set failure". In: *American Control Conference (ACC), 2011*. IEEE, pp. 1952–1957.

Roweis, Sam T and Lawrence K Saul (2000). "Nonlinear dimensionality reduction by locally linear embedding". In: *Science* 290.5500, pp. 2323–2326.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1988). "Learning representations by back-propagating errors". In: *Cognitive modeling* 5.3, p. 1.

Saito, Takaya and Marc Rehmsmeier (2015). "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets". In: *PloS one* 10.3, e0118432.

Sakoe, Hiroaki and Seibi Chiba (1978). "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE transactions on acoustics, speech, and signal processing* 26.1, pp. 43–49.

Salehi, Mahsa et al. (2016). "Fast memory efficient local outlier detection in data streams". In: *IEEE Transactions on Knowledge and Data Engineering* 28.12, pp. 3246–3260.

Salton, Gerard, Anita Wong, and Chung-Shu Yang (1975). "A vector space model for automatic indexing". In: *Communications of the ACM* 18.11, pp. 613–620.

Salvador, Stan and Philip Chan (2007). "Toward accurate dynamic time warping in linear time and space". In: *Intelligent Data Analysis* 11.5, pp. 561–580.

Santambrogio, Filippo (2015). "Optimal transport for applied mathematicians". In: *Birkäuser, NY*.

Santurkar, Shibani et al. (2018). "How does batch normalization help optimization?" In: *Advances in Neural Information Processing Systems*, pp. 2483–2493.

Saxe, Andrew M, James L McClelland, and Surya Ganguli (2019). "A mathematical theory of semantic development in deep neural networks". In: *Proceedings of the National Academy of Sciences* 116.23, pp. 11537–11546.

Schäfer, Patrick (2015a). "Scalable time series classification". In: *Data Mining and Knowledge Discovery*, pp. 1–26.

— (2015b). "The BOSS is concerned with time series classification in the presence of noise". In: *Data Mining and Knowledge Discovery* 29.6, pp. 1505–1530.

Schäfer, Patrick and Mikael Högqvist (2012). "SFA: a symbolic fourier approximation and index for similarity search in high dimensional datasets". In: *Proceedings of the 15th International Conference on Extending Database Technology*. ACM, pp. 516–527.

Schiavon, Michele et al. (2014). "Quantitative estimation of insulin sensitivity in type 1 diabetic subjects wearing a sensor-augmented insulin pump". In: *Diabetes care* 37.5, pp. 1216–1223.

Schirru, Andrea et al. (2012). "Learning from time series: Supervised aggregative feature extraction". In: *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. IEEE, pp. 5254–5259.

Schmid, Volkmar et al. (2010). "Pilot study for assessment of optimal frequency for changing catheters in insulin pump therapy—trouble starts on day 3". In: *Journal of diabetes science and technology* 4.4, pp. 976–982.

Schmidt, Ludwig et al. (2018). "Adversarially Robust Generalization Requires More Data". In: *arXiv:1804.11285*.

Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller (1997). "Kernel principal component analysis". In: *International Conference on Artificial Neural Networks*. Springer, pp. 583–588.

Scholkopf, Bernhard and Alexander J Smola (2001). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Scholkopft, Bernhard and Klaus-Robert Mullert (1999). "Fisher discriminant analysis with kernels". In: *Neural networks for signal processing IX* 1.1, p. 1.

Senin, Pavel and Sergey Malinchik (2013). "Sax-vsm: Interpretable time series classification using sax and vector space model". In: *2013 IEEE 13th International Conference on Data Mining*. IEEE, pp. 1175–1180.

Silva, Diego F and Gustavo EAPA Batista (2016). "Speeding up all-pairwise dynamic time warping matrix calculation". In: *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, pp. 837–845.

Sinkhorn, Richard (1964). "A relationship between arbitrary positive matrices and doubly stochastic matrices". In: *The annals of mathematical statistics* 35.2, pp. 876–879.

Soatto, Stefano (2007). "On the distance between non-stationary time series". In: *Modeling, Estimation and Control*. Springer, pp. 285–299.

Sonoda, Sho and Noboru Murata (2019). “Transport analysis of infinitely deep neural network”. In: *The Journal of Machine Learning Research* 20.1, pp. 31–82.

Sparck Jones, Karen (1972). “A statistical interpretation of term specificity and its application in retrieval”. In: *Journal of documentation* 28.1, pp. 11–21.

Stanforth, Robert, Alhussein Fawzi, Pushmeet Kohli, et al. (2019). “Are Labels Required for Improving Adversarial Robustness?” In: *arXiv preprint arXiv:1905.13725*.

Stöggl, T. et al. (2014). “Automatic classification of the sub-techniques (gears) used in cross-country ski skating employing a mobile phone”. In: *Sensors* 14.11, pp. 20589–20601.

Subasi, Abdulhamit (2007). “EEG signal classification using wavelet feature extraction and a mixture of expert model”. In: *Expert Systems with Applications* 32.4, pp. 1084–1093.

Subasi, Abdulhamit and M Ismail Gursoy (2010). “EEG signal classification using PCA, ICA, LDA and support vector machines”. In: *Expert Systems with Applications* 37.12, pp. 8659–8666.

Susto, G.A. et al. (2015). “Machine Learning for Predictive Maintenance: A Multiple Classifier Approach”. In: *Industrial Informatics, IEEE Transactions on* 11.3, pp. 812–820.

Susto, Gian Antonio and Alessandro Beghi (2016). “Dealing with time-series data in Predictive Maintenance problems”. In: *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*. IEEE, pp. 1–4.

Susto, Gian Antonio, Alessandro Beghi, and Seán McLoone (2017). “Anomaly detection through on-line isolation Forest: An application to plasma etching”. In: *Advanced Semiconductor Manufacturing Conference (ASMC), 2017 28th Annual SEMI*. IEEE, pp. 89–94.

Susto, Gian Antonio, Matteo Terzi, and Alessandro Beghi (2017). “Anomaly Detection Approaches for Semiconductor Manufacturing”. In: *Procedia Manufacturing* 11.Supplement C. 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy, pp. 2018 –2024. ISSN: 2351-9789. DOI: `https://doi.org/10.1016/j.promfg.2017.07.353`. URL: `http://www.sciencedirect.com/science/article/pii/S2351978917305619`.

Susto, Gian Antonio et al. (2013). “A predictive maintenance system for integral type faults based on support vector machines: An application to ion implantation”. In: *Automation Science and Engineering (CASE), 2013 IEEE International Conference on*. IEEE, pp. 195–200.

Susto, Gian Antonio et al. (2014). “Machine learning for predictive maintenance: A multiple classifier approach”. In: *IEEE Transactions on Industrial Informatics* 11.3, pp. 812–820.

Susto, Gian Antonio et al. (2016). “Supervised aggregative feature extraction for big data time series regression”. In: *IEEE Transactions on Industrial Informatics* 12.3, pp. 1243–1252.

Szegedy, Christian et al. (2013). “Intriguing properties of neural networks”. In: *arXiv:1312.6199*.

Tan, Chuanqi et al. (2018). “A survey on deep transfer learning”. In: *International Conference on Artificial Neural Networks*. Springer, pp. 270–279.

Tang, Jian et al. (2001). “A robust outlier detection scheme for large data sets”. In: *In 6th Pacific-Asia Conf. on Knowledge Discovery and Data Mining*. Citeseer.

Tang, Jian et al. (2002). “Enhancing effectiveness of outlier detections for low density patterns”. In: *Advances in Knowledge Discovery and Data Mining*, pp. 535–548.

Tang, Jiliang, Salem Alelyani, and Huan Liu (2014). “Feature selection for classification: A review”. In: *Data Classification: Algorithms and Applications*, p. 37.

Tenenbaum, Joshua B, Vin De Silva, and John C Langford (2000). "A global geometric framework for nonlinear dimensionality reduction". In: *science* 290.5500, pp. 2319–2323.

Thabit, Hood and Roman Hovorka (2016). "Coming of age: the artificial pancreas for type 1 diabetes". In: *Diabetologia* 59.9, pp. 1795–1805.

The Diabetes Control and Complications Trial and Epidemiology of Diabetes Interventions and Complications Research Group (2000). "Retinopathy and nephropathy in patients with type 1 diabetes four years after a trial of intensive therapy". In: *The New England journal of medicine* 342.6, p. 381.

TinyImageNet. https://tiny-imagenet.herokuapp.com/.

Tipping, M.E. (2001). "Sparse Bayesian learning and the relevance vector machine". In: *The journal of machine learning research* 1, pp. 211–244.

Tishby, Naftali, Fernando C. Pereira, and William Bialek (1999). "The Information Bottleneck Method". In: pp. 368–377.

Toffanin, Chiara et al. (2017). "Towards a Run-to-Run Adaptive Artificial Pancreas: In Silico Results". In: *IEEE Transactions on Biomedical Engineering*.

Tsay, Ruey S (2005). *Analysis of financial time series*. Vol. 543. John Wiley & Sons.

Tsipras, Dimitris et al. (2018a). "Robustness May Be at Odds with Accuracy". In: eprint: 1805.12152. URL: https://arxiv.org/pdf/1805.12152.

— (2018b). "There Is No Free Lunch In Adversarial Robustness (But There Are Unexpected Benefits)". In: *arXiv:1805.12152*.

— (2019). "Robustness May Be at Odds with Accuracy". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SyxAb30cY7.

Übeyli, Elif Derya (2009). "Combined neural network model employing wavelet coefficients for EEG signals classification". In: *Digital Signal Processing* 19.2, pp. 297–308.

Van Der Maaten, Laurens, Eric Postma, and Jaap Van den Herik (2009). "Dimensionality reduction: a comparative". In: *J Mach Learn Res* 10, pp. 66–71.

Veeravalli, Bharadwaj, Chacko John Deepu, and DuyHoa Ngo (2017). "Real-Time, Personalized Anomaly Detection in Streaming Data for Wearable Healthcare Devices". In: *Handbook of Large-Scale Distributed Computing in Smart Healthcare*. Springer, pp. 403–426.

Vega-Hernandez, O et al. (2009). "Increasing security in an artificial pancreas: diagnosis of actuator faults". In: *Health Care Exchanges, 2009. PAHCE 2009. Pan American*. IEEE, pp. 137–142.

Villani, Cédric (2008). *Optimal transport: old and new*. Vol. 338. Springer Science & Business Media.

Visentin, Roberto et al. (2015). "Circadian variability of insulin sensitivity: physiological input for in silico artificial pancreas". In: *Diabetes technology & therapeutics* 17.1, pp. 1–7.

Vishwanathan, SVN, Alexander J Smola, and René Vidal (2007). "Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes". In: *International Journal of Computer Vision* 73.1, pp. 95–119.

Vishwanathan, SVN, Alexander J Smola, et al. (2004). "Binet-Cauchy Kernels." In: *NIPS*, pp. 1441–1448.

Wang, Jin et al. (2013). "Bag-of-words representation for biomedical time series classification". In: *Biomedical Signal Processing and Control* 8.6, pp. 634–644.

Wu, Cinna, Mark Tygert, and Yann LeCun (2017). "Hierarchical loss for classification". In: *arXiv:1709.01062*.

Wu, J. et al. (2009). "Gesture Recognition with 3D Accelerometer". In: *Ubiquitous intell. and comp.* Pp. 25–38.

Xi, Xiaopeng et al. (2006). "Fast time series classification using numerosity reduction". In: *23rd international conference on Machine learning*. ACM, pp. 1033–1040.

Yang, Meng et al. (2011). "Fisher discrimination dictionary learning for sparse representation". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, pp. 543–550.

Ye, Lexiang and Eamonn Keogh (2009). "Time series shapelets: a new primitive for data mining". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 947–956.

— (2011). "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification". In: *Data mining and knowledge discovery* 22.1-2, pp. 149–182.

Yeh, Raymond A et al. (2017). "Semantic image inpainting with deep generative models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5485–5493.

Yeh, Yi-Ren, Zheng-Yi Lee, and Yuh-Jye Lee (2009). "Anomaly detection via oversampling principal component analysis". In: *New Advances in Intelligent Decision Technologies*. Springer, pp. 449–458.

Young, Tom et al. (2018). "Recent trends in deep learning based natural language processing". In: *ieee Computational intelligenCe magazine* 13.3, pp. 55–75.

Zagoruyko, Sergey and Nikos Komodakis (2016). "Wide residual networks". In: *arXiv:1605.07146*.

Zanon, M., G.A. Susto, and S. McLoone (2014). "Root Cause Analysis by a Combined Sparse Classification and Monte Carlo Approach". In: *IFAC World Congress*. Vol. 19. 1, pp. 1947–1952.

Zhang, Chiyuan et al. (2016). "Understanding deep learning requires rethinking generalization". In: *arXiv preprint arXiv:1611.03530*.

Zhang, Qiang and Baoxin Li (2010). "Discriminative K-SVD for dictionary learning in face recognition". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, pp. 2691–2698.