State of the Art

Warith HARCHAOUI

Contents

1 Introduction

Through perception and experience, each human being gathers data all the time and eventually process that data into representations that become our grasp onto the universe for interacting. We represent ideas and concepts for thinking and language and even get *pre*-representations of things we do not know. Although *artificial intelligence* has become a very popular expression, one can remark that representation is a key prerequisite for any of *processing* that we consider whether we talk about our intelligence as human beings our artificial intelligence for automatic machines. On one side of the spectrum of possibilities, excellent representations do not even require further processing and simple decision mechanisms on raw data are enough for good results. On the other side of the spectrum of possibilities, poor representations actually do require sophisticated expert-based or advanced statistical processing to output good results by taking into account domain-specific knowledge. During the 2010s, some major technological obstacles were crossed allowing the development of the so-called deep learning marked by the coming of a new era where the frontier between representation and processing becomes very blurry.

In many machine learning fields, defining a clear and sound objective is always key to produce good research but unfortunately, several so-called more intelligent tasks are impossible to define that well which leads us in this dissertation in statistics to insist on the statistical representation side rather than on the statistical processing side. Therefore, our main subject is representation: (i) of data among its groups in the first chapter, (ii) of data among its characteristics in the second chapter and (iii) of predictions with uncertainty in the third chapter. At the crossroads of three different fields namely statistics, deep learning and optimal transport that recently gained much scientific attention, our effort leverages that wealth of existing research to tackle representations in three contributions with different contexts: Wasserstein Clustering (DiWaC and GeWaC) to identify groups among data, Infinitesimal Wasserstein Maximal Distortion (InWaMaDi) to highlight relevant characteristics of data and Hypothesis of an Uncertainty Model (HUM) to estimate both supervised predictions and some uncertainty information.

2 Machine Learning Landscape

This dissertation focuses on neural networks (a. k. a. *deep* techniques) that enjoy recent tremendous sucess as a technology but much of the current described work is applicable to other tools like decision trees [Breiman, 2017] or kernel-based methods [Andrew, 2001]. That being said, *deep learning* changed scientists traditions about data: usually researchers separated feature-extraction and automatic-decision making tasks into two jobs. Nowadays, this frontier becomes merely a blurred line as best systems are built by doing feature extraction and automatic decision simultaneously along layers (hence the "deep" adjective as more layers give more sophisticated systems). Mixing feature extraction and automatic decision in layers amplified by much more computation power than ever let neural networks become fancy again. The whole machine learning scientific community beyond those who uses neural networks refer to these low-financial-support periods as "artificial intelligence winters". In the 1990s and 2000s, even in a major conference named *Neural Information Processing Systems*, popular methods like kernel-based ones were more popular than neural networks at a time when the keyphrase "deep learning" was merely confidential. As announced in the seminal keynote at the International Joint Conference on neural networks in 2011, neural networks were back again in the performance leaderboards top methods thanks to recent hardware considerable computational improvements. Then, between 2011 and 2014, neural networks algorithms beat state-of-the-art records in several fields such as image processing, computer vision, speech recognition, machine translation with almost only artificial neural networks scientists and without researchers from specific domain expertise as explained by Ng [2013] which gave the surprising hope for an increasing ease in a growing list of applications.

During the last decade, a pleasing *end-to-end* paradigm emerged and says that systems should not be trained sequentially (or even independently) but rather simultaneously as a whole [Ng, 2018]. This intuitive belief that consists in training several layers of a neural network at once is widespread

for better empirical results. Unfortunately, doing an end-to-end training also means dealing with black boxes as intermediate neural networks layers that are difficult to interpret (and are even not identifiable whereas other less efficient methods still provide interpretation ease). End-to-end training seems to be encouraged for better than other known styles of training empirically and especially in Deep Learning [Bojarski et al., 2016].

Nevertheless, this statement must be handled with care for pragmatic reasons beyond industrial scalability and loss of interpretable modularity as Glasmachers [2017] points out there are also some other effects: feeding a deep neural network with the concatenation of raw data and some non-deep-learning algorithms outputs is often hard to beat. For example, in video recognition [Schmid, 2013, Crasto et al., 2019], it is recommended to augment the raw video voxels input with optical flow (which is a processed version of the same raw video pixels but for motion estimation). Indeed, because of the *data starvation* phenomenon (a. k. a. over-fitting), using off-the-shelves pretrained algorithms is a simplistic form of transfer learning combining knowledge (and sometimes data) from the current and from the previous tasks. Thus, pragmatically, it is sometimes useful not to follow the end-to-end-training approach just for the sake of it: sequentially trained and/or optimized modules can work very well and still provide the easiest interpretation for what each module does. One too naive end-to-end-training approach would end up with a black box trained from scratch.

In this dissertation, there is a will to emphasize our scientific need to crack in deep learning black boxes (end-to-end or not) because that's how better data understanding gets in, beyond automatic decisions.

2.1 Dimensions

In this dissertation, we will consider large scale settings so we must be specific about what is considered *large*. Throughout the machine-learning-related fields, we can consider:

Cardinality N, the number of data points for train;

Dimensionality D, the dimensionality of one data point;

Output dimension K, the dimensionality of the automatic output decision (number of classes in classification, the output space dimension for regression and beyond, the approximate number of nodes in a grammar tree, or number of atoms of a chemical molecule graph...).

as emphasized by Harchaoui [2013] in the concept of *machine learning cuboid*. For each edge of this cuboid, we have direct optimization implications for feasible computations and best results so far:

- $N \gg 1$ stochastic-gradient-based optimization algorithms are more suitable than in-memory alternatives because we only need a few data points (mini-batches) at the same time per iteration;
- $D \gg 1$ some further analysis should be conducted: dimensionality reduction and domain-specific knowledge must be used at once for fighting against the *curse of dimensionality* phenomenon;
- $K \gg 1$ one-class-vs-rest strategies are preferred rather than one-class-vs-one strategies for computational reasons. Indeed in a one-class-vs-rest strategy, we only need to combine K decisions separating each class with all the rest whereas in a one-class-vs-one strategy, we would have $\frac{K \times (K-1)}{2}$ decisions separating all combinations of classes pair.

Throughout this work, we are mainly interested in large N and large D data configurations without considering large K issues. For example, in clustering settings, K should be small because otherwise it defeats the data analysis purpose: having too many clusters does not help human beings to understand data. Although the large N and large D case has already been successfully investigated in the supervised classification context, at the beginning of this work (in 2016) little research had been conducted but we observe that this key preoccupation is finally entitled to scientific attention today.

Along these three dimensions of data analysis in machine learning, this Ph.D. dissertation proposes new (or revisited) representations: (i) simplifying the *cardinality axis of N* thanks to clustering in our

first contribution, (ii) an attempt to better understand data at a coordinate level for a local *dimensionality axis of D* relevance assessment in our second contribution, and (iii) re-interpreting the *output axis of K* through uncertainty estimation in our third contribution.

2.2 Epistemology

Epistemology is the theory of knowledge [Newman, 2018, Ahmad, 2003]. In particular, in machine learning, one epistemology has consequences on our beliefs, opinions, justifications and finally our scientific methodology in our studies. Several epistemological ways to describe machine learning exist and for this dissertation we choose one with probabilistic perspectives [Murphy, 2012] because of its ease of sophisticated interpretation for insights. We believe data is coming from a phenomenon that we call *Nature* that we represent by an idealized probabilistic distribution associated with a random and often multivariate variable (or a pair or a tuple of this random multivariate variables). In practice, we consider datasets as extracts of Nature. An (annotated) dataset is some collection of independent realizations that are identically (sampled) distributed from what we call Nature. Of course, this statement is falsifiable [Bernard, 1898] and maybe counter-intuitive but computer science, statistical learning, machine learning, data science and all these young sciences mixing mathematics and programming are just a few decades old, compared to more established several centuries old (or even millenia old) fields such as mathematics, biology, physics, chemistry, medecine etc.

Mathematically, it is convenient to choose that epistemology (rather than an other one) in order to introduce the notion of generalization capabilities of machine learning systems. Indeed, with other epistemology, datasets have a higher status and generalization becomes ill-defined (with the *ad hoc* notion of training error and testing generalization error). In this work, we accept that datasets empirical distributions (sum of Dirac distributions) are approximated and noisy versions of an idealized (probably smoother) distribution.

2.3 Different kinds of machine learning

Generally, for a given project, the job of a machine learning scientist (or data scientist depending on the name given by economic trends) is decomposed in two phases often in a loop: (i) training time (previous to or interleaved with a validation time) to match/imitate/reproduce the phenomenon in the presence of groundtruth information (labels, reward) for learning a model, (ii) test time (or execution time of the system we have just built) without access to groundtruth information because we are using our trained model. Following pioneers in machine learning [LeCun, 2015], we can roughly separate the machine learning landscape in three depending on what is accessible during training, validating and testing times: first supervised learning, second unsupervised learning and third reinforcement learning as this dissertation can find applications in all of these three machine learning fields.

For the purposes of notation, univariate functions are here generalized to the multivariate case by applying the associated univariate function to each entry and then concatenating everything such that the function output has the same shape as the input. For example, $\mathbf{z} \in \mathbb{R}^K$ is a vector whose $K \in \mathbb{N}^*$ coordinates $\mathbf{z}^{(k)}$ are indexed by k, then $\log(\mathbf{z}) = \left[\log(\mathbf{z}^{(1)}), \ldots, \log(\mathbf{z}^{(k)}), \ldots, \log(\mathbf{z}^{(K)})\right]^\top$.

2.3.1 Supervised Learning

Nature provides a pair of (input, output) random variables $(\mathbf{x}, \mathbf{y})$.

$$(\mathbf{x}, \mathbf{y}) \sim \text{Nature} \tag{1}$$

collected in a training labelled dataset (or *Nature extract* as stated above):

$$\begin{gathered}\text{dataset} = (\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_i, \mathbf{y}_i), \ldots, (\mathbf{x}_N, \mathbf{y}_N) \\ \text{where each } (\mathbf{x}_i, \mathbf{y}_i) \text{ is a realization of } (\mathbf{x}, \mathbf{y}) \sim \text{Nature}\end{gathered} \tag{2}$$

On the one hand, input $\mathbf{x}$ often represents a question in various forms such as: a vector of numbers (categories, integers or floating decimal numbers), an image or a video (made of pixels, voxels in channels and beyond [Ponce and Forsyth, 2011]), a sound (its waveform or its time-frequency representation [Li et al., 2016, Mallat, 2008]), a gene (its ATGC or RNA-seq representation [Barillot et al., 2012]), a chemical molecule (its 3D graphical representation [Zaslavskiy, 2010]) etc. On the other hand, output $\mathbf{y}$ represents the answer of the question $\mathbf{x}$ in two main classes of problems: regression in which we deal real numbers and classification dealing with categories and integers. Of course, these kinds of separations are limited but somewhat useful to describe the main problems.

Many supervised learning problems share the same kind of optimization objective:

$$\min_{\mathcal{F}} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\text{Nature}} \left(\ell\big(\mathbf{y}, \mathcal{F}(\mathbf{x})\big)\right) \tag{3}$$

where $\ell\big(\mathbf{y}, \mathcal{F}(\mathbf{x})\big)$ measures the discrepancy of predicting $\mathcal{F}(\mathbf{x})$ from an input $\mathbf{x}$ instead of the ground truth label $\mathbf{y}$. This equation Eq. (3) behaves like an aggregation of errors (when $\mathbf{y} \neq \mathcal{F}(\mathbf{x})$) summed up into one value (the lower, the better) over data. In layman's terms, $\ell(\mathbf{y}, \mathcal{F}(\mathbf{x}))$ is *how much the system is punished for a mistake during training* and is preferably zero for no error: a perfect $\mathbf{y} = \mathcal{F}(\mathbf{x})$ scenario. More precisely, statistical learning becomes the task of finding parameters $\boldsymbol{\theta} = \boldsymbol{\theta}_{\mathcal{F}}$ of function $\mathcal{F}$ that has the right structure (tree, random forest, linear or kernel-based support vector machines, neural networks etc.) that minimizes $\mathcal{L}$:

$$\begin{gathered} \min_{\boldsymbol{\theta}_{\mathcal{F}}} \mathcal{L}(\boldsymbol{\theta}_{\mathcal{F}}) \\ \mathcal{L}(\boldsymbol{\theta}_{\mathcal{F}}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\text{Nature}} \left(\ell\big(\mathbf{y}, \mathcal{F}(\mathbf{x})\big)\right) \end{gathered} \tag{4}$$

Indeed, the formulation of Eq. (4) has the merit of generalizing almost all supervised learning problems.

Beyond the scope of this dissertation, there is a considerable amount of scientific works about regularization notably inherited from Lagrangian optimization [Boyd and Vandenberghe, 2014] and similar techniques. In a nutshell, our Eq. (3) is still valid to fit in this kind of research by simply replacing the current function ℓ by function $\tilde{\ell}$:

$$\tilde{\ell}(\mathbf{y}, \mathcal{F}(\mathbf{x})) \triangleq \ell(\mathbf{y}, \mathcal{F}(\mathbf{x})) + \lambda\Omega(\mathcal{F}) \tag{5}$$

where Ω gives the predictor function $\mathcal{F}$ some desirable properties (see the books of Bonnans et al. [2003] and of Boyd and Vandenberghe [2014] for further details) with a relative importance given by $\lambda \in \mathbb{R}_+$ in order to provide generalization capabilities coping with the fact that we only have access to a limited training dataset instead of Nature itself in practice.

In summary, supervised learning is finding a function $\mathcal{F}$ that maps $\mathbf{x}$ to $\mathbf{y}$ based on a training dataset assuming that such an idealized function $\mathcal{F}^*$ exists (sometimes we only need $\mathcal{F}^*$ to only be a relation and not necessarily a function):

$$\mathbf{y} \simeq \mathcal{F}^*(\mathbf{x}) \tag{6}$$

In order to build an estimator $\hat{\mathcal{F}}$ of the desired decision function $\mathcal{F}^*$, we usually solve an optimization problem:

$$\hat{\mathcal{F}} = \min_{\mathcal{F}} \mathcal{L}(\mathcal{F}) \tag{7}$$

where the loss function $\mathcal{L}$ is a proxy of all errors that we want to ideally minimize in one value $\mathcal{L}(\mathcal{F})$ measuring all aggregated discrepancies between the ground truth $\mathbf{y}$ and prediction $\mathcal{F}(\mathbf{x})$.

In practice, we only have access to a limited amount of annotated $(\mathbf{x}, \mathbf{y})$ data that we call training data to *fit* our prediction function $\hat{\mathcal{F}}$, so the $\mathcal{L}(\mathcal{F})$s are estimated by approximations $\hat{\mathcal{L}}(\mathcal{F})$ as if the available data was all the data in the universe: almost as if the Nature smooth distribution was replaced by the dataset empirical distribution. The hope consists in saying that at test time, for a new and unseen input $\mathbf{x}$ coming from the same $(\mathbf{x}, \mathbf{y})$ distribution as in training time but where the true output $\mathbf{y}$ is unknown, we can predict an estimated output $\hat{\mathbf{y}} = \hat{\mathcal{F}}(\mathbf{x})$ that is close to the true output $\mathbf{y}$.

Learning is possible because at training time, we have access to both the input $\mathbf{x}$ and output $\mathbf{y}$ in a dataset. Back to our philosophical considerations about epistemology, we may consider that the supervised learning task consists in *compressing* the relationship between the training pairs $(\mathbf{x}, \mathbf{y})$ in a fitted predictor $\hat{\mathcal{F}}$ that once trained, one only needs $\mathbf{x}$ to recover a lossy version of $\mathbf{y} \simeq \hat{\mathcal{F}}(\mathbf{x})$. During the "compression process of learning" we accept some loss in information that we sacrifice to get better compression rate in general non-lab conditions. This way, we can study many machine learning tools such linear dot products, tree, random forest of trees, non-linear kernel evaluations, vanilla neural networks, convolutional or recurrent neural networks etc. and their respective algorithms, computations and structures with the same compression-flavored point of view.

In supervised learning we have:

$\mathbf{x}$ and $\mathbf{y}$ at training time
$\mathbf{x}$ without $\mathbf{y}$ at testing time that we estimate

Looking at supervised learning as a compression problem is interesting for understanding recurring trade-off through out this scientific literature: models should be sophisticated enough to recover outputs information from inputs without too much loss of information (complexity and number shoud go up) while still being sufficiently sober (i. e. not too sophisticated,(complexity and number goes down) otherwise generalization capabilities dramatically drop down while compression is given up.

2.3.2 Unsupervised Learning

In unsupervised learning, Nature does not provide any more information than the data $\mathbf{x}$ itself.

$$\mathbf{x} \sim \text{Nature} \tag{8}$$

This machine learning field is useful for data analysis that has more broader scientific purposes than the industrial applications of supervised learning.

$\mathbf{x}$ without any labeled $\mathbf{y}$ information that we still estimate to get structure from data *for the sake of interpretable knowledge discovery*

Imitating data (Generative Adversarial Networks GANs for example) [Goodfellow et al., 2014] and clustering [Jain, 2010] are two unsupervised learning tools to exhibit data analysis as a fundamentally intelligent tool for scientists, *intelligent* etymologically meaning from latin understanding the underlying structure of data. Unsupervised learning is often ill-posed which remains a mystery because the same automatic tasks can be evaluated subjectively many times by different human beings with still some consistency (e. g. for clustering) which gives hopes for improvements. More mathematically, indeed, we use the Hadamard definition [Maz'ya and Shaposhnikova, 1999] for a well-posed problem and we understand that all these three points cannot apply in clustering objectives optimization:

1. a solution exists,
2. the solution is unique,
3. the solution's behaviour changes continuously with the initial conditions.

In this dissertation, clustering is tackled in a chapter as the computation of the maximal optimal transports between groups (e. g. clusters) while leveraging the GANs scientific literature in terms of numerical tools. Furthermore, another chapter is trying to make use of enhanced Wasserstein distance among distributions for relevant features weighting of data without explicit supervision from any task but rather the *likeness* of each point with the remaining dataset points that we express with the notion of *worst* optimal transport once again thanks to its powerful mathematics and algorithmics machineries.

2.3.3 Reinforcement Learning

Reinforcement Learning [Sutton and Barto, 2018] is an area of machine learning about how agents (say robots) sequentially observing environment inputs $\mathbf{x}$ from Nature could take the best sequence of actions $\mathbf{y}$ in order to maximize some untimely reward $\mathbf{r}$ also given by Nature and not necessarily after each action while maintaining a state representation $\mathbf{s}$ (or $\mathbf{z}$) of both history and environment.

$$(\mathbf{x}, \mathbf{r}) \sim \text{Nature, but } \mathbf{r} \text{ is not always given (i. e. we often have } \mathbf{r} = 0\text{)} \tag{9}$$

For example, playing automatically chess, Go, cards are famous applications associated with artificial intelligence victories in controlled settings face to expert human beings.

In unsupervised learning, we have:

> $\mathbf{x}$ but no supervised action $\mathbf{y}$ is given, only some reward $\mathbf{r}$ once in a while during training time
>
> $\mathbf{x}$ with some reward $\mathbf{r}$ once in a while at testing time and we estimate the best sequence of actions $\mathbf{y}$

In reinforcement learning, taking into account uncertainty estimation is certainly helpful for the distentangling the recurring exploration / exploitation dilemma [Sutton and Barto, 2018]. This opens up a potentially large range of possible applications for our contribution dealing with uncertainty.

3 Neural Networks

Since their introduction in computer science [Rosenbaltt, 1957], artificial neural networks loosely inspired by the biological neurons did not stop fascinating researchers until today. In a nutshell, a neural network implements a function $\mathcal{F}$ from $\mathbb{R}^D$ to $\mathbb{R}^K$ by the composition of $L \in \mathbb{N}^*$ layers (or sub-functions) $(\mathcal{F}_\ell)_{\ell=1,\dots,L}$ of the form:

$$\begin{array}{rrcl} (\forall \ell \in [\![1, L-1]\!]) & \mathcal{F}_\ell & = & a \circ \text{Linear}_\ell \\ & \mathcal{F}_L & = & \text{Linear}_L \end{array} \tag{10}$$

where:

- The non-linear element-wise function a called activation is often chosen among the hyperbolic tangent function, the positive part function (or rectified linear unit ReLU), the sigmoid function (or logit function).
- The Linear_ℓs functions are matrix-vector product linear operators in the form of:

  $$(\forall \mathbf{x} \in \mathbb{R}^{i_\ell}) \;\; \text{Linear}_\ell(\mathbf{x}) = \mathbf{M}_\ell \mathbf{x} + \mathbf{b}_\ell \tag{11}$$

  for $\mathbf{M}_\ell \in \mathbb{R}^{o_\ell \times i_\ell}$ and $\mathbf{b}_\ell \in \mathbb{R}^{o_\ell}$ with $(i_\ell, o_\ell) \in \mathbb{N}^* \times \mathbb{N}^*$ and $(\forall \ell \in [\![1, L-1]\!]) \;\; o_\ell = i_{\ell+1}$ ($i_1 = D$ and $o_L = K$)

Thanks to the universal approximation theorem of Hornik [1991], we only need mild conditions on a (unbounded and non-constant) for the set of functions expressed in this form to be dense over the set of Lebesgue-integrable functions.

This means that each time, we have an objective function to minimize over a set a function, there exists a neural network $\mathcal{F}$ that is able to approximate the optimal solution arbitrarily well. In terms of computer science, this is good news because instead of minimizing over a set of functions, we can reasonably minimize over a set of parameters approximating that function we are looking for. Thus, we transformed a functional optimization problem into a numerical optimization problem over the matrices $\mathbf{M}_\ell$s, the biases $\mathbf{b}_\ell$s but also the number of layers L and the input/output parameters (i_ℓ, o_ℓ). The initial enthusiasm provoked by this statement was dampened through decades because this is only an existence theorem that does not provide a way in itself to find these parameters.

Moreover it turns out that the statistical estimation of these parameters is difficult due to the overfitting phenomenon [Scholkopf and Smola, 2001] (a. k. a. *data starvation* which is a complementary metaphor). Optimization is also slow (especially in the 1960s, 1970s, 1980s and even 1990s compared to nowadays).

Recently, neural networks became suddenly more plausibly useful since storage and computations were getting much faster and cheaper. Indeed, on top of a dramatic computational speed improvement, fast random data access is also crucial for realistic real-world applications. During the 1990s, even with industrial-level quality results, scientists in neural networks for machine learning did not catch the whole Research community world wide attention at first. As data storage and collection problems have been nicely solved thanks to the decrease of hardware's costs (and thus the increase of available computational power especially with the rise of CPUs and GPUs parallelism and distribution), increase of data access speed and software solutions (like HDFS [Shvachko et al., 2010] and Spark [Zaharia et al., 2010]) that all appeared in an era around the 2000s called *Big data*. In the 2010s, these tools for manipulating data were key for large-scale training and execution of machine learning engines [Castelluccio, 2017]. Another factor of scientific success is the extensive use of world-wide open source repository which pioneered in terms of reproducibility and best practice sharing. On the theoretical part, the mild conditions of the universal approximation theorems [Cybenko, 1989, Hornik, 1991, Gao and Jojic, 2016] only gives approximation ability up to our statistical estimation ability (hence the need of big cardinality datasets compared to dimensionality). Otherwise, neural networks notoriously suffer from data starvation (a. k. a. overfitting) and the neural networks need appropriate structures, computations and optimization procedures to inject enough knowledge into the systems in order to get the tremendous sucess we benefit today. Still, in spite of these recent and great improvements, there is a lack of solid theoretical grounds (especially compared to its Reproducing Kernel Hilbert Space RKHS counterpart) and many open questions remain unanswered as of today although we can mention the works of Vidal et al. [2017] and Arora et al. [2017].

3.1 Input Data

Thanks to the universal approximation theorem presented previouly, it is reasonable to parse several kinds of input data to see how neural networks can be fed. Indeed, neural networks universal approximation theorems do only provide an existence result of a desired function but no explicit way to get or estimate it. Even with this important theorem, scientists sill have to work to build an adequate deep learning structure to cope with the estimation problem of an almost unreachably ideal neural networks parameters. Even if we had a procedure that could reach that ideal neural networks, this would be fitted on training data only which is not a guarantee for good results on unseen yet data.

Historically [Rosenbaltt, 1957], input data $\mathbf{x} \in \mathbb{R}^D$ is a vector fed to a function implemented by a one-layer perceptron: a composition of a matrix-vector product and an element-wise sigmoid function. At the same time, stochastic optimization tools were revisited, implemented for large scale settings with the associated theoretical background provided by the Robbins-Monro theorem [Robbins and Monro, 1951] (we will come back on it later). The idea of composition for more sophisticated neural networks came fast with the introduction of multilayer perceptron [Rosenblatt, 1961] along with the back-propagation algorithm which gracefully adapts the differenciation chain rule for efficient algorithms (the *forward* step is the evaluation of the neural network function and the *backward* step is the computation of its gradient with respect its parameter). Nowadays, thanks to much engineering progress, more and more artificial intelligence promises are kept.

Since the dawn of the 1990s, new kinds of information media became available on computers storage systems with increasing sophistication and some dramatic research-to-product type of improvements emerged:

Text Word processor softwares rapidly replaced typewriters and made natural language processing possible and one can cite the old Reuters text dataset [Hayes and Weinstein, 1990] for example. In terms of artificial intelligence, it seems that this medium along with genetics data are the most difficult one;

Image Likewise, digital recording of photographs made the *singleton* dataset of Lena [Roberts, 1962] in

image processing grow from only one to several thousands images size datasets with MNIST [LeCun et al., 1989b] and to millions images size datasets such as ImageNet [Fei-Fei, 2010] and beyond. The associated tremendous research progress made possible real-world applications in everyday applications;

Sound Early speech recognition systems also benefit from datasets collection since one of the oldest: TIMIT [Zue et al., 1990], even the music processing got its own MusicNet dataset [Thickstun et al., 2018];

Video the recent 2018 YouTube 8 millions videos dataset [Abu-El-Haija et al., 2016] seem promising for same kind of quantitative-qualitative upward gap and improvements as image in a near future.

There is an interesting hypothesis to maybe understand why the text medium is so hard to manipulate within statistical frameworks compared to the other ones although it has been the first to be digitized with enormous ever-growing quantity data: close features in the other media are more clearly dependent (two neighboring pixels in images and even voxels in videos are highly dependent like consecutive sound samples are) and this dependence fades away with longer horizons but unfortunately text features (words or even letters) have stronger and longer range interactions that suggests to see them through grammar in spite of the scientifically obsolete and harmful but widespread saying from Frederick Jelinek in 1985:

> "Every time we fire a phonetician/linguist, the performance of our system goes up."

Combining strong statistical tools and linguistics is probably the best alternative for future natural language representations.

For all these different media (or information supports), a major research pattern can be analyzed: most data contain redundant information so discarding stuttered information is useful in order to manipulate the relevant infortmation only. For example, spatial data like images, have highly dependent close pixels and thus intuitively small-sized convolution kernels seem appropriate to decorrelate the redundant information. Another example is sequence data or smooth time series data where neighboring data (with respect to the sequence or temporal axes) should also be decorrelated thus 1D-convolutions [Zhang et al., 2015] or recurrence seem appropriate as much as recurrent neural networks can [Murakami and Taguchi, 1991, Hochreiter and Schmidhuber, 1997] as described in Fig. 1.

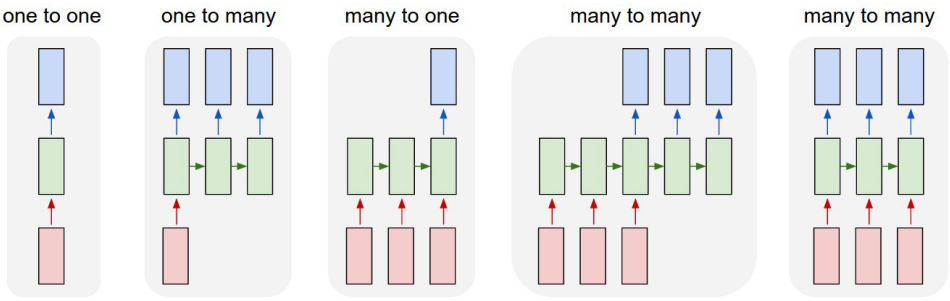

Figure 1: Several input/processing/output scenarios from standard neural networks to recursive ones. Red: Input, Green: Processing, Blue: Output. From left to right: one (input) to one (output) vanilla neural network, one to many like in image captioning (one image to many words in a sentence), many to one like in sentiment analysis (many words in a sentence to a category of mood), and the (delayed or not) many to many case like in langauage translation (many words of a sentence in one language to many words of a new sentence in a different language). Diagrams taken from the pedagogical blog of Andrej Karpathy: `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`

This huge research effort allowed for large scale applications (both in terms of training size or cardinality N and data size or dimensionality D and in the end the output size K). Meanwhile, image and sound processing were increasingly developping methods using convolutions when LeCun et al.

[1989b,a] applied those convolutions for image classification with both iterated compositions and backpropagation which turned out to be also useful for sound and even text processing [LeCun and Bengio, 1995]. For time series, there is still a scientific debate among the deep learning community about whether recurrent or convolutional neural networks (which abreviate to RNN and CNN respectively). Meanwhile the sequential nature of words enumeration in text is tackled as time series but there is probably more hidden structure yet to get from grammar (as pioneering work from Socher et al. [2011] pointed out since 2011).

These combined breakthroughs allowed ambitious real-world applications throughout the 1990s and the 2000s. Today, new constibutions with improved engineering, structure, optimization and regularization tools are still referring to early works with even homage (e. g. GoogLeNet by Szegedy et al. [2015] referring to LeNet by LeCun et al. [1989b]). Image and Speech recognition is embedded in many everyday products, even video recognition begins to have reliable industrial applications. Natural language processing also gets impressive translation results[1] but there is still room for improvements. For all these different media, it seems that the same phenomenon occurs: unleashing clean dataset with thorough engineering effort astonishingly helps the scientific community to bring back high quality prototypes and ultimately products. Indeed, if we measure the time delay between a dataset release and available products and the best example is the AdaBoost implementation for face detection with Haar features (MIT-CMU frontal faces dataset [Sung et al., 1998] and prize-wining paper [Viola and Jones, 2001] less than 3 years after). In computer vision, the same phenomenon occured with more and more available large cardinality datasets and thus ready-to-use products as Fig. 2 shows.

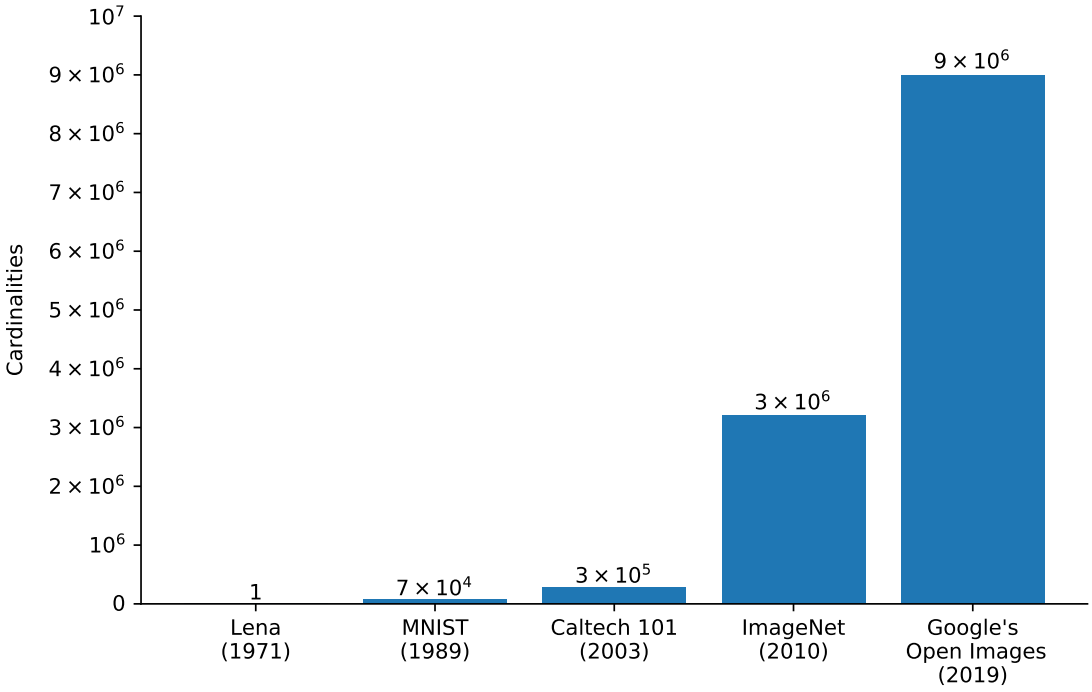

Figure 2: Images Datasets Explosion

3.2 Output Data and Functions Properties

Now we present a table of tips to adapt unconstrained vanilla neural networks functions to constrained custom ones without the need of tediously maintaining the constraints (which is not recommended for stochastic gradient descent optimization, the weapon of choice to face large scale datasets according to the neural networks literature [LeCun et al., 1998]).

[1] `https://www.deepl.com/press.html`

Mathematically, the universal approximation theorem allows to parse a very large class of function (more precisely we only need the Lebesgue-integrability [Hornik, 1991]):

$$\mathbf{x} \in \mathbb{R}^D \text{ and } \mathcal{F}(\mathbf{x}) \in \mathbb{R}^K \tag{12}$$

and Table 1 enumerates many kinds of functions and their corresponding implementations thanks to the unconstrained original neural network function $\mathcal{F}$.

Property	Implementation
Positivity	$\exp(\mathcal{F}(\mathbf{x}))$ or $\mathcal{F}(\mathbf{x})^2$ or even $\max(0, \mathcal{F}(\mathbf{x}))$
Boundness between m and M ($m < M$)	$m + (M-m)\sigma(\mathcal{F}(\mathbf{x}))$ where $\sigma(\mathbf{z}) = \frac{1}{1+\exp(-\mathbf{z})}$ which is related to the SoftMax function when $K = 2$
Probability Vector (i. e. in K-dimensional simplex)	$\text{SoftMax}(\mathcal{F}(\mathbf{x}))$ where $\text{SoftMax}(\mathbf{z})_k = \frac{\exp(\mathbf{z}_k)}{\sum_{\ell=1}^K \exp(\mathbf{z}_\ell)}$ which is related to the multilogit model [Hastie et al., 2005] and the logsumexp trick
Positivity and (semi)-definiteness matrix	$\mathcal{C}(\mathbf{x}) \times \mathcal{C}(\mathbf{x})^\top$ where $\mathcal{C}(\mathbf{x})$ is a lower triangular free matrix with positive diagonal entries (even bounded for more stability in practice) which is related to the Cholesky decomposition [Golub and Van Loan, 2012]
1-Lipschitz function	Online power iteration on each matrix-vector product inside the neural network that implements the function $\mathcal{F}$ which is described in the spectral normalization work by Miyato et al. [2018]
Bijection (one-to-one function)	Composition of layers such as $\left[\mathbf{x}_{[:d]}^\top, \left(\boldsymbol{s}(\mathbf{x}_{[:d]}) \times \mathbf{x}_{[d:]} + \boldsymbol{t}(\mathbf{x}_{[:d]})\right)^\top\right]^\top$ (with *pythonic* indexation of coordinates) where $\boldsymbol{s}$ and $\boldsymbol{t}$ are regular neural network functions. This technique also gives to the log-determinant of the Jacobi matrix without too much computation burden thanks to the original paper of Dinh et al. [2017]
Recursive Function	If t is time and $\mathbf{y}_0$ an initialization: $\mathbf{y}_{t+1} = \mathcal{G}(\mathbf{y}_t, \mathcal{F}(\mathbf{x}_t))$ which is pedagogically well presented by Karpathy [2015] and Chakraborty et al. [2014] with improved LSTMs variants by Hochreiter and Schmidhuber [1997] and GRUs [Cho et al., 2014]

Table 1: Implementations for several types of Functions with respect to their inputs nature and functional properties

4 Optimal Transport

The optimal transport research field has proven to be crucial in redefining our modern world as we know it, across a stunningly wide range of applications, since its French birth in the XVIIIth century [Monge, 1781], with scientists from very different backgrounds and application areas such as Rabin et al. [2012] in image processing, Courty et al. [2017] in near-general data domain adaptation, Abouchar [1970] for airports management, decisive World War II military battle victories [Smolinski, 1962], breakthrough innovation in modern economy [Galbraith, 2019] and flabbergasting futuristic industrial revolutions such as semi-automatic objects generative design [Shu et al., 2019]. After this long one-sentence celebration, we now briefly review the basics of such a prolific mathematical offspring.

4.1 Formulations

In a data space $\mathcal{X}$ equipped with a metric c, we want to measure a distance between two *piles* of data that is related to that metric. We are willingly using the vague word *pile* because thanks to the notion of Dirac distibutions we have access to both smooth densities, empirical distributions and a wide variety of distributions in general. Mathematically, we thus manipulate two distributions $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ with associated two variables $\mathbf{x} \sim \boldsymbol{\mu}$ and $\mathbf{y} \sim \boldsymbol{\nu}$ both living in $\mathcal{X}$. On those distributions, we want to compute the quantity $W_c(\boldsymbol{\mu}, \boldsymbol{\nu})$ measuring how much different the *piles* $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are. To that end, three equivalent formulations exist for that same quantity:

Monge Formulation

$$W_c(\boldsymbol{\mu}, \boldsymbol{\nu}) = \inf_{T_*(\boldsymbol{\mu})=\boldsymbol{\nu}} \mathbb{E}_{\mathbf{x}\sim\boldsymbol{\mu}}\big[c\left(\mathbf{x}, T(\mathbf{x})\right)\big] \tag{13}$$

where $T_*(\boldsymbol{\mu})$ denotes the push forward of $\boldsymbol{\mu}$ by transport map T. For real understanding, we refer the reader looking for details to academic textbooks on Probability such as *Random Measures, Theory and Applications* [Kallenberg, 2017].

Wasserstein Formulation

$$W_c(\boldsymbol{\mu}, \boldsymbol{\nu}) = \inf_{\gamma\in\Gamma(\boldsymbol{\mu},\boldsymbol{\nu})} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\gamma}\big[c\left(\mathbf{x}, \mathbf{y}\right)\big] \tag{14}$$

where $\Gamma(\boldsymbol{\mu}, \boldsymbol{\nu})$ denotes the set of couplings γ based on the two distributions $\boldsymbol{\mu}, \boldsymbol{\nu}$: the collection of all probability measures on $\mathcal{X} \times \mathcal{X}$ with marginals $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ in order to maintain two coupling properties:

$$\forall \mathbf{y} \in \mathcal{X},\ \mathbb{E}_{\mathbf{x}\sim\boldsymbol{\mu}}\left[\gamma(\mathbf{x}, \mathbf{y})\right] = \boldsymbol{\nu}(\mathbf{y}) \tag{15}$$

and

$$\forall \mathbf{x} \in \mathcal{X},\ \mathbb{E}_{\mathbf{y}\sim\boldsymbol{\nu}}\left[\gamma(\mathbf{x}, \mathbf{y})\right] = \boldsymbol{\mu}(\mathbf{x}) \tag{16}$$

Kantorovich-Rubinstein Formulation (for the L_2 euclidean distance cost: $c(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$)

$$W(\boldsymbol{\mu}, \boldsymbol{\nu}) = \sup_{\mathcal{C}\in\text{Lip-1}} \mathbb{E}_{\mathbf{x}\sim\boldsymbol{\mu}}\left[\mathcal{C}(\mathbf{x})\right] - \mathbb{E}_{\mathbf{y}\sim\boldsymbol{\nu}}\left[\mathcal{C}(\mathbf{y})\right] \tag{17}$$

where Lip-1 is the 1-Lipschitz functions set (from $\mathcal{X} \subset \mathbb{R}^D$ to $\mathbb{R}$). More general formulas exist in *Real analysis and Probability* by Dudley [2018], but changing the euclidean distance cost is possible up to how difficult redefining the Lipschitz property of $\mathcal{C}$ is.

As a scientist apprentice, one can notice that for producing research work, it seems that the Monge formulation is more amenable to intuition, the Wasserstein formulation is more suitable for probabilistic and geometric perspectives and the Kantorovich-Rubinstein formulation is more convenient for devising algorithms thanks the decoupling of $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ in the formulas which makes computations easier (simple expectations that a usual Monte Carlo estimation can handle).

4.2 Algorithms

Since the middle of the XXth century, three major algorithmical tools emerged:

1. Hungarian Discrete Method [Kuhn, 1955]
2. Entropy-Regularized Sinkhorn Fixed-Point Method [Cuturi, 2013, Genevay, 2019]
3. Wasserstein Generative Adversarial Networks [Goodfellow, 2016]

but we are well aware that with a certain amount of pragmatism it is beyond the scope of this humble state of the art to present the great mathematical achievements in optimal transport. Daring to write a summary of such a huge mathematical and on-going research field is unsettling and we refer the reader to three great references for best and rather exhaustive optimal transport overview: (i) *Optimal Transport for Applied Mathematicians* by Santambrogio [2015] on the general scientific culture side, (ii) *Computational Optimal Transport* by Peyré et al. [2019] on the statistical and programming side, (iii) *Optimal Transport: Old and New* by Villani [2008] on the probabilistic and theoretical side.

Historically, Kuhn [1955] found a cubic complexity algorithm to solve an optimal transport in the discrete case which was internationally widespread as the so-called *Hungarian Method*. Linear cost optimization of one-to-one assignments between two sets of elements is reccuring in many real-world applications as briefly enumerated above. Indeed, once a pairwise assignment cost matrix is given, minimizing the associated cost sum with respect to the best possible assignment map boils down to the ticking of one matrix entry per row and per column (with some additional dummy entries for coping with the rectangular matrix case i. e. different cardinalities for the two sets at hand). Beyond this seminal successful attempt to cast optimal transport as a linear problem, some other works pushed the analysis further with network flows [Ahuja et al., 1989], with graph theory angle [Goldberg and Tarjan, 1989], then Dynamic Programming mixed with fluid mechanics reasoning came in with Benamou and Brenier [2000] for improved computational speed. Special discrete-continuous distributions cases were also efficiently tackled by Mérigot [2011] with some exceedingly fast convergence thanks to Lévy [2015]. First in industrial logistics, these approaches were used in a surprisingly wide range of applications from e. g. worker to work assignments, airplane to airport assignments, to communication protocol load balancing systems etc.

Recently, Cuturi [2013] revisited entropy regularized transport to efficiently solve almost the same optimal transport problem with the Sinkhorn iterative fixed-point-type algorithm, which is especially handy when polynomial complexity is not realistic in large scale settings. The idea is that they are willing to trade some approximated *optimal* transport due to transport entropic regularization for realistic speed and doable computations. Surprisingly, even exploiting the entropy-regularized properties of the Sinkhorn (and thus *non-optimal*) transport itself has value in many applications such as robust finance [De March, 2018], ranking [Vert], photo album summarization [Liu et al., 2020]... This is explainable because traditionally, regularizations schemes are meant to make numerical and stability problems vanish. That fixed-point Sinkhorn theorem gives extremely fast convergence rate of transport entropy regularized over Wasserstein distances computations: less than a dozen of iterations are enough in practice. This approach on top of dramatic computational accelerations makes it indispensable both for theoretical analysis and for many real-world applications. In spite of diligent progress for Wasserstein Generative Adversarial Networks (as we will describe later), the work accomplished by Genevay [2019] still presents Sinkhorn-based techniques as a great mathematical and numerical alternative for efficient optimal-transport-related solutions in machine learning.

In order to imitate high dimensional data, the principle of the milestone work of Goodfellow et al. [2014] about Generative Adversarial Networks (GAN) is prototypically new for a fascinating worldwide series of research papers. As illustrated in Fig. 3, from a pseudo-random generator, we can sample some low dimensional noise that is transformed thanks to a variable generator function to get generated data within the original data space. The role of the critic function is to estimate a divergence between the real data and the generated data distributions and the generator's role is to minimize it. The most common metaphor for this mainstream press acclaimed technique is considering the generator as a forger trying to fool the detective embodied by the critic within an adversarial objective. The detective wants to distinguish generated and real data and the forger wants to produce generated

data that are indistinguishable when compared to real data. More mathematically, it turns out that there is indeed a link between that min-max optimization and a Nash equilibrium as emphasized by Fedus et al. [2017].

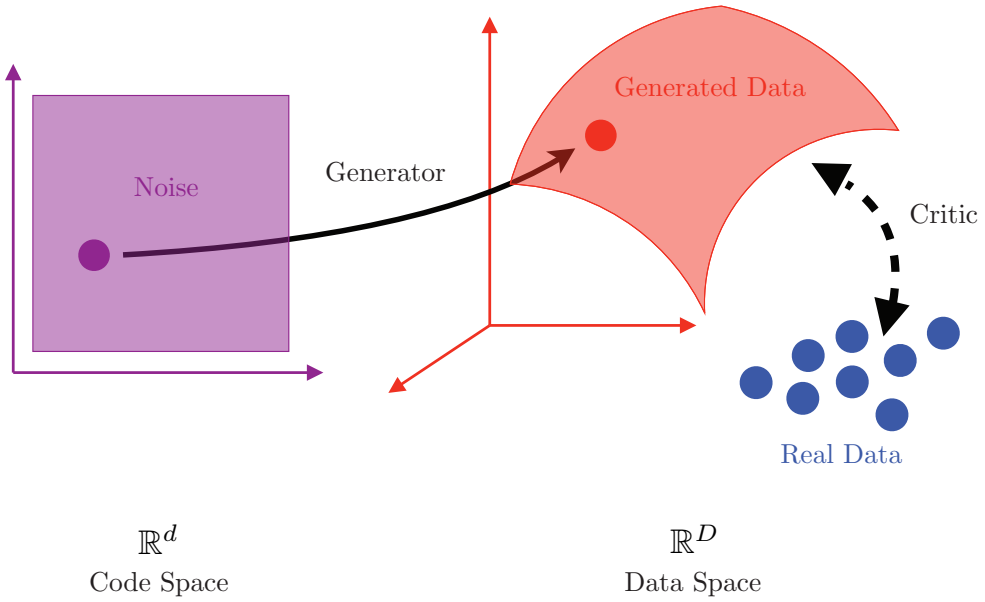

Figure 3: GAN Principle: Imitating Data thanks to some Artificial Low Dimensional Noise in Purple from a Code Space ($\mathbb{R}^d$) transformed thanks to a Generator function (or forger) into Generated Data in Red living in the Data Space (($\mathbb{R}^D$)) so that they are supposed to be close to the Real Data in Blue thanks to the Critic function (or detective) – Adapted from `https://optimaltransport.github.io`

Following the seminal Generative Adversarial Networks (GANs) work of Goodfellow et al. [2014], Arjovsky et al. [2017] used the Kantorovich-Rubinstein formulation in order to imitate data extending GAN from Jensen-Shannon divergence minimization to Wassertein distance minimization between generated and real data distributions which paves the way for revisiting optimal transport in the context of unsupervised learning. Answering the high research expectations for GANs, the recent contribution of Miyato et al. [2018] called spectral normalization had a tremendous impact. Indeed, revisiting the power iteration numerical recipe [Press et al., 2007] at each linear or convolutional steps to elegantly enforces the Lipschitz property required by the Kantorovich-Rubinstein duality allows variation constraints withtout unstable stochastic gradient projection techniques beyond Wasserstein distances and optimal transport. Until the spectral normalization technique, neural networks had a tendency to implement function that are not regular and enforcing the Lipschitz property (i. e. constraining the variations of the functions implemented this way) gives a beneficial regularization effect beyond Wasserstein distance estimation.

This spectral normalization technique gave so much optimization stability that facing unheard-of large scale settings is made possible and provides the extraordinary images imitation results of the BigGAN approach [Brock et al., 2019]. This engineering achievement is also convincing thanks the already-mastered residual convolutional neural networks [He et al., 2016] ResNet tool coming from the supervised image classification research. As a matter of fact, it is fair to say that ResNet [He et al., 2016] gave the fascinating ability to neural networks of handling recursively raw, moderately pre-processed and highly pre-processed data at each layer by *short-cutting* the traditional successive-layers structure thanks to additional cross-layer connections. All combined, it made BigGAN [Brock et al., 2019] impressive rendering results possible, neatly fighting against the curse of dimensionality effect on the learned parameters side confronted with exceedlingly large scale unsupervised conditions for both data cardinality N (number of images in the dataset) and dimensionality D (number of pixels for the image high resolution) under unsupervised conditions.

5 Representations

Knowing how to represent data is understanding data and the underlying structure beneath it (and *vice versa*). As is, data has in general too much dimensionality to be plotted ($D > 3$) and finding a useful projections seems to interleave regular dimensionality reduction tehniques and clustering (i. e. the task of building groups as we will see later).

Since 2012, several researchers identified what large scale settings for computations and memory meant not only for applied mathematics but also for the worldwide economy describing it at the *Big Data* era: Among them Zikoupoulos and Eaton [2016], Peters [2012], Jordan [2013]. Consider N, the number of elements of a database (the cardinality), D the size of each element (the dimensionality), then we observe two training conditions or regimes for machine learning algorithms:

Big data regime $\frac{N}{D} \gg 1$ Statistical theorems behave nicely but software programming was difficult before data storage and computations speed dramatically improved (even on the software side for parallelization)

Small data regime $\frac{N}{D} \ll 1$ or $N \simeq D$ As we will see, the *curse of dimensionality* make things very difficult in terms of statistics but computations are easy.

In practice, software and hardware issues of the big data regime have been solved in the 2000s and the beginning 2010s at an industry-quality level. Domain specific and statistical expertise are still required for small data regime. Indeed with images for example, hierarchical convolutions organized layers in neural networks has a decorrelating effect which reduces the impact of the huge dimensionality of data: intermediate results (called feature maps) are still very big on the first layers but the number of parameters has been substantially decreased thanks to the small-sized convolution kernels. For other domains, applying Convolutional Neural Networks (CNN) did work but mastery of each domain must not be ignored. In other words, injecting enough knowledge into an automatic system is basically diminishing the dimensionality (i. e. removing redundancy with respect to the task at hand) which is good news towards coping with the curse of dimensionality: this transforms a difficult small data regime into an easier big data regime.

5.1 Big data and neural networks

We previouly established that a convincing decrease of dimensionality D thanks to the appropriate representation tools is key for good empirical results in machine learning. Now it is time to explain how to optimize an objective function under large *cardinality* conditions. Since the 1950s, some early work mixing statistics and optimization, Robbins and Monro [1951] allowed large cardinality training dataset thanks to stochastic optimization. One interesting aspect of the Robbins-Monro theorem is that the objective (nor the full-gradient) does not need to be evaluated directly anymore but only a biased-free estimate of its gradient with respect to the learned parameters. In the end, the large scale constraint is relieved because only randomly picked mini-batches of data are needed instead of the whole dataset during the optimization. This early mathematical finding paved the way for contemporary large scale learning fulfilled ambitions [Bonnans et al., 2003] beyond deep learning. Several scientific avenues have been taken with sometimes sophisticated algorithmical tools [Bertsekas, 1997] with new programming context (e. g. distributed systems [Hendrikx et al., 2019] or limited memory systems [Defazio et al., 2014]).

In a nutshell, the Robbins-Monro theorem allows to manipulate an idealized loss function (over all the data of the universe) without the need to compute its values nor its gradients with respect to its parameters as long as one can provide a biased-free estimator of the loss function gradient needed for the stochastic gradient descent optimization scheme. Surprisingly, it turns out that artificial neural networks actually look like biological neural networks for learning (but much less for its structure and running behavior inspite of 1950s predictions). Indeed, stochastic gradient descent follows a Hebbian rule[2]:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \mathbf{f}_t \tag{18}$$

[2]Donald O. Hebb was an influential neuro-psychologist from the 1950s

where t is an iteration index, $\boldsymbol{\theta}$ the parameters that are learned, α is a learning rate (often constant) and finally $\mathbf{f}_t$ is an incremental progress computed thanks to a measured error. Numerically, $\mathbf{f}_t$ is the gradient of an objective $\mathcal{L}$ to minimize:

$$\mathbf{f}_t = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t) \tag{19}$$

which can be approximated by a biased-free estimate of $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t)$ according to the Robbins-Monro theorem. For a general objective to minimize such as $\mathcal{L}(\boldsymbol{\theta}_{\mathcal{F}}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim \text{Nature}} \left(\ell\left(\mathbf{y}, \mathcal{F}(\mathbf{x})\right)\right)$ the hebbian rule has a replaced stochastic gradient $\hat{\mathbf{f}}_t$ instead of the "true" one $\mathbf{f}_t$:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \hat{\mathbf{g}}_t \tag{20}$$

where

$$\hat{\mathbf{f}}_t = \frac{1}{B} \sum_{b=1}^{B} \nabla_{\boldsymbol{\theta}} \left(\ell(\mathbf{y}_{i_b}, \mathcal{F}(\mathbf{x}_{i_b}))\right) \tag{21}$$

and $i_b \sim \mathcal{U}_{\mathbb{N}}(1, N)$ is an uniform index parsing the N-cardinality training dataset in mini-batches of size B.

It is noteworthy to recall that there is a simple case where the best learning rate is given in closed form: the online estimation of a mean random multivariable which is related to the least squares problem:

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \frac{1}{t+1}(\mathbf{x}_{t+1} - \boldsymbol{\mu}_t) \tag{22}$$

From a practitioner point of view, large cardinality problems have been solved both on the software/hardware side and on the optimization side. The dimensionality issues still remain and is in essence more specific to each application we tackle.

5.2 The Curse of Dimensionality

Enhancing data structure in the data representation is an efficient way to cope with the curse of dimensionality issue that we focus on in this section. As said above, in images and speech, convolutions provided enormous improvements in terms of accuracy for the given task. Word ordering structure is today considered as standard since the rise of embeddings techniques with *word2vec* [Mikolov et al., 2013] and variants even with the recent *BERT* method. The fundamental idea is to transform observed occurrences of words ordering on large corpora into a regression problem providing relevant words representations into real-valued vectors. The situation is much more difficult in genomics despite worldwide sincere attention [Barillot et al., 2012]. Indeed 1 human is DNA-represented by $D \simeq 10^{12}$ nucleotides and the worldwide population is $N \simeq 10^9$ which corresponds to a small data regime once again. Even in diagnostics problem *healthy v. s. ill* detection or classification problem, in theory a balanced and worldwide-sized annotated DNA-dataset is not enough for classification nor interpretation which is quiet an embarassing (theoretical) scenario. We did not yet leverage enough desirable and yet unknown DNA structural information among those nucleotides to get an easier big data regime.

Now we know that high cardinality is not a problem anymore if dimensionality is not too high keeping the problem under the big data regime. In small data regime when dimensionlity is too high, devising algorithms is difficult because of the well-known *Curse of Dimensionality*[3] that we mentioned earlier. To explain that phenomenon dubbed the *Curse of Dimensionality*, a classic example considers the volume of a sphere of unit radius in dimension D which follows

$$V(D) = \frac{\pi^k}{(k)!} \text{ if } D = 2k \text{ is even, or } V(D) = \frac{2(k!)(4\pi)^k}{(2k+1)!} \text{if } D = 2k+1 \text{ is odd} \tag{23}$$

[3]The keyphrase *Curse of Dimensionality* was first mentioned by Bellman [1957] to describe the need of efficient algorithmic tools like Dynamic Programming [Dasgupta et al., 2008] to efficiently explore huge discrete solutions spaces, historically later it also concerned many kinds of large data spaces

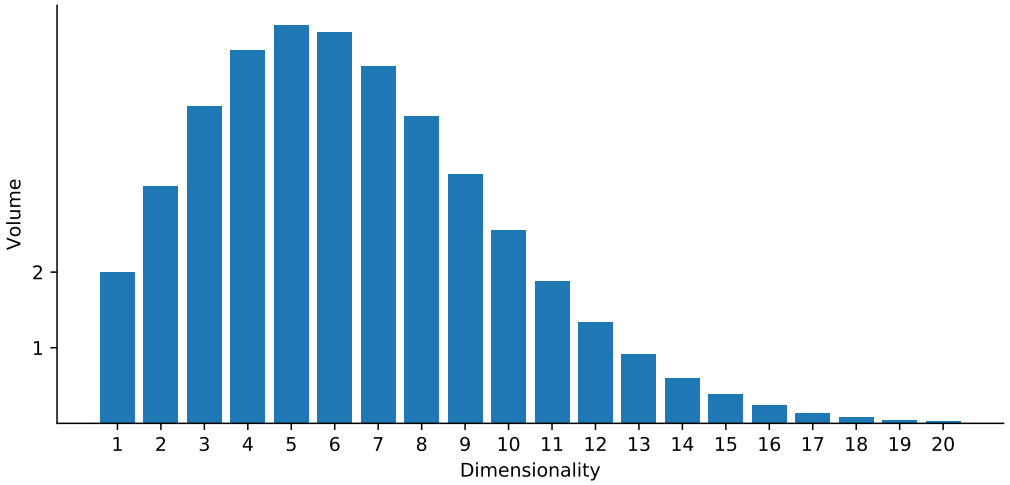

Figure 4: Volume of an euclidean sphere of unit radius in dimension D

In Fig 4, first we can see that volume $V(D)$ reaches its maximum for $D = 5$ and right after, that same volume $V(D)$ dramatically decreases towards 0 as D grows ($\lim_{D\to\infty} V(D) = 0$) which is not intuitive and rather deceiving because of how we experience spatial neighborhood notions in our 2D and 3D living environment as human beings (we would expect that volume to grow indefinitely as it does for $D = 1, 2, 3, 4$). For example, in low dimensionality regimes ($D = 1, 2$ or 3), if a point lies close to the origin inside an unit ball neighborhood, then the volume to exhaustively parse for finding it is reasonably big ($V(D) \simeq 2, 3.14$ or 4.18 respectively). But for high dimensionality regimes (e. g. $D = 30$), the corresponding volume to parse gets extremely small ($V(D) =\simeq 2 \times 10^{-5}$) which is unsettling: we would think that looking for a point inside a higher dimensional sphere would be larger but this is not true. In the end, this means that rudimentary notions like distances or even similarities behave unexpectedly in such high dimensionality data regimes. Another pedagical example considers the ratio $R(D)$ between a 0.9-radius and 1-radius balls' volumes ($R(D) = 0.9^D$). In Fig. 5, that $R(D)$ ratio also exhibits an embarassing phenomenon with respect to our intuition: as the dimensionality D increases, the volume ratio $R(D)$ goes to zero ($lim_{D\to\infty} V(D) = 0$) which means that in high dimensionality regimes the *orange 0.1-peel* occupies almost all the *entire orange* volume in layman's terms.

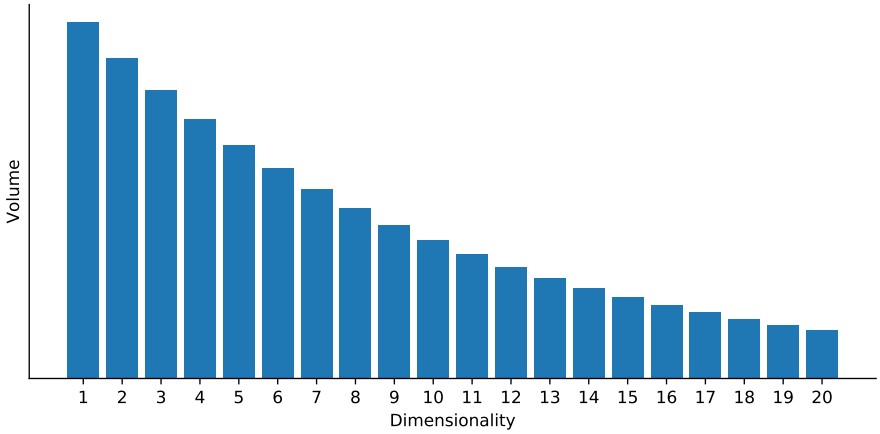

Figure 5: 0.1-peel ratio $R(D)$f or an euclidean sphere of unit radius in dimension D

For pedagogical reasons, we may present a metaphor that we dubbed the "crinkled paper in a room". Indeed, in general, real data distribution seem to behave like a crinkled sheet of paper with low intrinsic manifold dimensionality (say $d = 2$) inside of a large space of dimensionality D (say a

$d = 2$ crinkled sheet of paper living in a room of $D = 3$ dimensions to help our $3D$-living creatures intuition). In this metaphor, revealing the data structure and focusing on the independent variables ruling the data boils down to how we *iron* or *un-crinkle* that sheet of paper to explicitly have access to the real degrees of freedom parsing the data.

Interestingly, following this metaphor helps us understand than the GAN approach is doing the opposite: starting from a low dimensional uniform law (the ironed sheet of paper) transformed by the generator into a higher dimensional and much more sophisticated law. GAN optimization is basically learning how to *un-iron* or *crinkle* a clean sheet of paper into something close to the real data manifold.

Unfortunately, in terms of measures theories, neither the Riemann nor Lebesgue measures allocate weight to that sheet of paper (crinkled or not, it is respectively not defined and zero) which may lead to explore new theoretical research avenues and maybe studying other probability foundations like maybe not yet sufficiently explored in machine learning based on the Hausdorff measure [Abbott and Rogers, 1999] or using probability freed from measure theory like the acclaimed attempt of Breiman [1992]. Another way to cope with this difficulties is to abandon the notion of probability distributions to embrace energy-based models as initiated by LeCun et al. [2006] which require less assumptions to model data and allows to generalize a distribution into an energy (through a neg-exponentiation from an energy analogy from Physics but without the constraint of having a unit *measure* over space).

On the kernel-based techniques [Shawe-Taylor et al., 2004] side during the 1990s and 2000s, the dimensionality problems disappear because once a kernel similarity matrix is built, one does almost not need the data anymore to operate analysis (with supervised kernel-based support vector machines or unsupervised spectral clustering). With kernels, dimensionality problems are avoided thanks to the kernel similarity matrix. All dimensionality-wise considerations are relegated to the crucial kernel definition. Unfortunately, such techniques are limited due to the inherent quadratic memory complexity of such pairwise structures implied by the similarity matrix.

Meanwhile, on the Model-based side, naive approaches do not succeed to achieve good results because of that high dimensionality: they ultimately loose their specific model selection capabilities because of over-parametrization to match that high data dimensionality. Indeed, in reasonable dimensionality conditions, the main advantage of such techniques is their ease for probabilistic interpretation and model selection. Over-parametrization (or over-fitting) can be seen as a *data starvation* phenomenon: a large number of parameters to fit would require a huge amount of data to get reliable estimations which in practice leads to poor performance. We see that once again, the interesting factor is the ratio $\frac{N}{D}$ of cardinality N over D and not one of them without considering the other one.

In the context clustering of large scale dimensionality, parcimonious and cluster-wise representations [Bouveyron and Brunet-Saumard, 2014] circumvent these high dimensionality problems and still keep the appealing probabilistic properties of model-based clustering without sacrificing accuracy. One can remark that in the supervised classification literature, sparsity (and even structured sparsity [Jenatton, 2011]) also did cope with dimensionality problems that are similar in essence.

Model-based clustering algorithms are popular because they are renowned for their probabilistic foundations and their flexibility [Duda et al., 2012]. Indeed, even for non-statisticians, the possibility to output meaningful probabilities is intuitive and principled. The main drawback of mixture-based and model-based methods for clustering is the lack of richness (in Kleinberg's sense see [Kleinberg, 2015] but we will come back on it later) due to necessary distribution assumptions that may not be necessarily true for real data which justifies our attempt to alleviate this limitation thanks to the functional expressivity of neural networks.

One fundamental machine learning hypothesis is recurring in the literature [Murphy, 2012, Duda et al., 2012, Bishop, 2006]: real data live in a low-dimensional (of dimension D) manifold in a much higher dimensional space (of dimension D and $d \ll D$). A classic pedagogical example consists of the independent and uniform sampling of each pixel of an image: there is no realistic chance to produce a convincing photograph! This means that even sophisticated mathematical object such as photographs lie on a manifold of lower intrinsic dimensionality than the number of pixels multiplied by the number of channels. In a reverse fasion, this has been confirmed by the DCGAN work of Radford et al. [2015] that is able to generate $D = 3 \times 256 \times 256 \simeq 2 \times 10^5$ convincing DCGAN

images from a random uniform variable made of $d = 100$ independent coordinates). Thanks to this low-manifold-dimensionality hypothesis for data in mind, it is reasonable to investigate some dimensionality reduction techniques.

5.3 Dimensionality Reduction

At the beginning of the XXth century, Principal Component Analysis (PCA) was invented [Pearson, 1901]. This technique finds an optimal linear (or affine) projection with respect to compression/decompression quadratic reconstruction error. This algorithm gave birth to two more recent ones: (i) its kernelized extension [Schölkopf et al., 1998] (euclidean distances can be expressed with dot products that are in turn replaced by kernel evaluations in a Reproducing Kernel Hilbert Space RKHS following the well-known *kernel trick*) and (ii) auto-encoders [Kramer, 1991, Bourlard and Kamp, 1988, Vincent et al., 2010] which replaces compression and decompression by one neural network each.

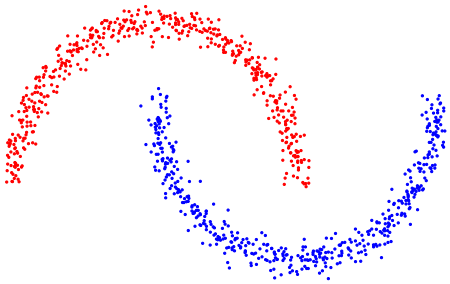

Figure 6: "Two Moons" Toy Dataset

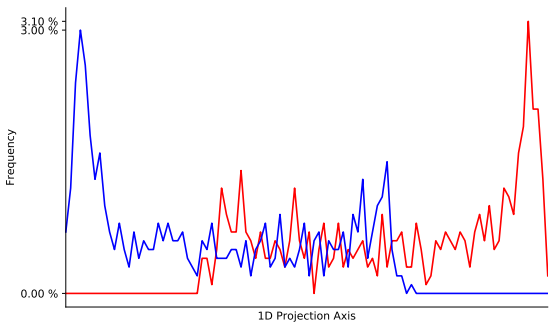

Figure 7: Two Moons Projection through PCA into 1D

Fig. 7, 8 and 9 show PCA, (Gaussian) kernel-PCA and AE dealing with non-linearly-structured yet simple 2D distributions shown in Fig. 6. It turns out that (up to the Gaussian parameter of our kernel-PCA), the non-linearity improvements of PCA in two different variants namely kernel-PCA and AE does help *ironing the crinkled distribution of interest* (to follow our metaphor in section 5.2).

Dimension reduction approaches such as principal components analysis (PCA) or even autoe-encoders (AE) [Vincent et al., 2010] may help for clustering but, as is, are not designed with a clustering mindset which causes poor results in practice. These global dimension reduction techniques are pragmatic but ignore information which is discriminant for separating clusters. Indeed, clusters are usually living in different sub-spaces between clusters if there exist. Back in the original data space, there is no reason to find an easy-to-find common linear sub-space that is discriminant enough to separate all the classes at the same time. For example, if clustering is taken into account while reducing the dimensionality, then one solution could be to divide the reduced space into as many zones as clusters such that they do not overlap while still reducing the dimensionality. Thus, we

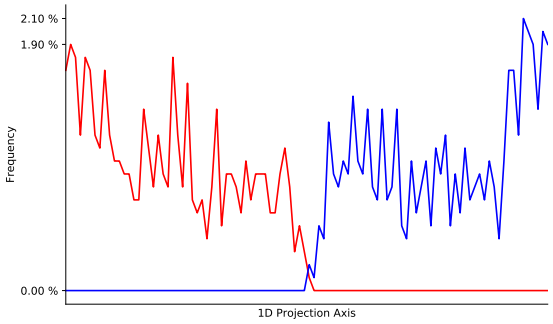

Figure 8: Two Moons Projection through kernel-PCA into 1D (with Gaussian Kernel)

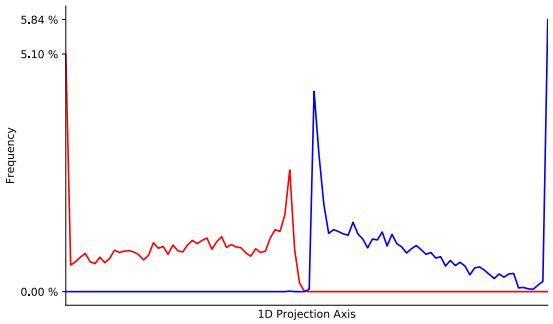

Figure 9: Two Moons Projection through an Auto-Encoder into 1D

avoid generic approaches because they cannot afford by themselves to capture these subtleties in the data. One must combine dimensionality reduction and clustering. Clustering could be looked at as an extremely simplified version of the data by just keeping the index of the cluster the data belong to.

The high dimensionality clustering literature [Bouveyron et al., 2007] tends to show that clustering and dimensionality should be done a the same time (*i.e.* in an end-to-end fashion) and not sequentially. Indeed, on the one hand, doing clustering first for huge dimensionality data is computationally difficult for obvious reasons and also statistically difficult because of the *curse of dimensionality*. On the other hand, doing dimensionality reduction first looses hidden cluster-wise information about data. One major issue of this work is precisely trying to tackle this "chicken or egg" problem.

Vanilla Auto-encoders alone do not allow to specify a precise probabilistic structure for a low-dimensional representation. This limits their combination with model-based clustering techniques. Furthermore, optimizing an auto-encoder and a Gaussian mixture generally implies the use of a trade-off hyper-parameter to combine these two objectives. This hyper-parameter is possibly hard to tune as cross-validation is not an option in our unsupervised settings as no validation score can be used by definition.

The problem of learning representations from data in an unsupervised manner is a long-standing problem in machine learning [Bengio et al., 2013, LeCun et al., 2015]. Principal Components analysis (PCA) and auto-encoders (AE) which can be seen as non-linear extension of PCA [Baldi and Hornik, 1989] have been used for representing faces [Turk and Pentland, 1991] or to produce a hierarchy of features [Chan et al., 2015]. Other techniques have been used such as sparse coding [Mairal et al., 2008] where the representation of one image is a linear combination of a few elements in a dictionary of features. More recently Bojanowski and Joulin [2017] learned features unsupervisedly by a procedure that consists in mapping a large collection of images to noise vectors through a deep convolutional neural networks.

Clustering and dimensionality reduction are interleaved. The importance of finding a suitable representation for unsupervised tasks was first highlighted by Chang [1983], who showed that

embeddings based on principal component analysis were often unfit for clustering purposes so we suggest the idea of learning both clustering and dimensionality at the same time in an end-to-end deep learning fahsion. In a more model-based literature [Bouveyron et al., 2019], combining clustering and dimensionality reduction simultaneously also proved more successful than separating dimensionality reduction and clustering sequentially, which in turn, was already more successful than doing only one of them for both results. This means that both deep learning and Bayesian literatures tend to show a certain symbiosis between clustering and dimensionality reduction towards data analysis and understanding.

In the context of linear embeddings (that offers dimensionality reduction), the main approach was to combine linear discriminant analysis with the k-Means (k-Means) algorithm (DisKMeans) [De la Torre and Kanade, 2006] or more generally a Gaussian Mixture Model (Fisher-EM) [Bouveyron and Brunet, 2012]. Much less research is available in relation to non-linear embeddings. Archambeau and Verleysen [2005] however proposed to use manifold learning in combination with GMM. Combining clustering with representation learning has been done with deep learning techniques in the past. An early attempt was explored by Trigeorgis et al. [2014] who used a deep semi-non-negative-matrix-factorization (NMF) model to specifically factorize the input into multiple stacking factors which are initialized and updated layer by layer with k-Means on the last layer.

Neural networks have proven successful in the context of supervised classification and even regression [Goodfellow et al., 2016]. Indeed, their ability to transform data such that the frontiers between classes are hyperplanes in the classification setting have made them very popular. In spite of the non-convexity of their optimization scheme, today, they are superior to convex machineries such as Support Vector Machines even for kernelized ones in almost every domain in Computer Vision and sound processing for example. The idea of having learned features has already been tackled by Chen [2015] used Deep Belief Networks together with maximum-margin clustering. Wang et al. [2016] jointly optimized a sparse coding objective and a clustering loss. Eventually, all these recent approaches have been empirically outperformed by auto-encoders' style machineries.

When it comes to compressing data while limiting loss of reconstruction information, auto-encoders have proved efficient [Vincent et al., 2010]. Briefly, an auto-encoder is a neural network made of two parts: (i) the *encoder* maps the data in a low-dimension space, (ii) the *decoder* maps them back to the original space. An auto-encoder is trained to reconstruct the data in the original space (usually in a least squares fashion but it could be any differentiable metric). At the end, if the reconstruction error is low, then codes resulting from the encoder (also called "bottleneck") have compressed data without loosing too much information (because by construction, it is possible to rebuild data from codes thanks to the decoder). The main assumption behind this technique is that the input data space of high dimensionality contains structure that could be successfully embedded in a lower-dimensionality manifold [Alain and Bengio, 2014, Sonoda and Murata, 2016] and the code space plays that embedding role.

Generative Adversarial Networks (GAN) [Goodfellow et al., 2014] establish a min-max game between a generator neural network on one side and a discriminator or critic neural network on the other side in order to generate data (from random noise) that the critic cannot distinguish from the real data. From that influential work emerged Adversarial Auto-Encoders (AAE) by Makhzani et al. [2015], Wasserstein Auto-Encoders (WAE) by Tolstikhin et al. [2018] and Adversarially Learned Inference (ALI) by Dumoulin et al. [2017]. In a few words, thrice are turning an auto-encoder into a generative model. They are trained in different ways that put an arbitrary fixed prior distribution in the code space. For the clustering chapter, we were initially inspired by these approaches with learned mixture distribution instead of a fixed prior one.

6 Dissertation Outline

This thesis is illustrating the process of learning representations using neural networks and optimal transport through three applications:

Clustering *joint work with Pierre-Alexandre Mattei, Andrés Almansa and Charles Bouveyron* It is about unsupervisedly representing large-scale datasets in groups. This offers data tools to get a better

intimate knowledge of the data with whereas the usual deep learning supervised classification algorithms do not so easily unless tedious manual annotation is already *paid for* at least on training data;

Unsupervised Feature Importance It consists in analyzing data at a coordinates level with a wide of applications from pure data understanding to background/foreground image segmentation in an unsupervised manner (the only remaining supervision being a pile of images containing the same semantic class of content). In this work, we only propose an attempt to accomplish that desirable goal and we provide a sound theoretical framework to do it;

Prediction with Uncertainty We insist on a better interpretation of supervisedly trained neural networks output in terms of uncertainty (especially for classification probabilities) towards a simple yet efficient way to improve uncertainty estimation in such supervised learning scenarios. In this contribution, sensitive applications can be a little more reliably envisioned when simple industrial constraints or more complex health, security and even justice issues are involved.

At the end of this mauscript, we provide an appendix *Generative Adversarial Networks Initialization with Auto-Encoders*. This is an heuristic for practical initialization of GAN training with tips revisiting pre-training traditions from the 1990s but for contemporary machine learning tools.

In this *state of the art* chapter, we introduced the challenges related to thrice learning representations, optimal transport and neural networks and the next chapters will be devoted to applications in clustering, unsupervised feature importance extraction and supervised uncertainty estimation thanks to the aforementioned tools. Some ongoing and future work are envisioned as a conclusion and an appendix presents practical yet effective techniques for training GANs that we gathered from experience.

6.1 Clustering

Clustering is one of the oldest unsupervised learning task [Jain, 2010]. Clustering [Duda et al., 2012] is the task of making groups without the need of any manual annotations. Along with dimensionality reduction, clustering is a desirable goal in data analysis, visualization and is often a preliminary step in many algorithms for example in computer vision [Ponce and Forsyth, 2011] and natural language processing [Goldberg, 2017]. Clustering and more generally data analysis does not only consist in pre-processing steps, it is about helping us (as human beings) understanding the underlying structure of data at hand.

Meanwhile, the computer vision field has recently witnessed major progress thanks to end-to-end deep-learning systems since the seminal work of LeCun et al. [1990] and more recently of Krizhevsky et al. [2012]. Most of the work however has been carried out in a supervised context. Our effort leverages that wealth of existing research but in an unsupervised framework.

While optimal transport [Villani, 2008] have gained recent attention especially for generating data (*i.e.* imitating data) in large scale settings (large both in terms of dataset cardinality N and dimensionality D) with Generative Adversarial Networks (GAN) originated by Arjovsky et al. [2017] and Sinkhorn divergences by Genevay [2019], we chose to ignore imitating capabilities and just use this literature to algorithmically manipulate Wasserstein distances. For example, this considerable amount of anterior work gives us a significant ease for optimization with helpful tools such as stochastic gradient descent.

The purpose of this research is to build a linear-complexity algorithms that use non-linear embeddings into code spaces. Indeed, in the clustering literature, one can distinguish two kinds of clustering algorithms with respect to their computation and memory complexity as function of the cardinality N. On one side, we have linear algorithms such as k-Means (k-Means) and Gaussian Mixture Models (GMM), which usually work directly on the data (*i.e.* without any medium such as embeddings and transformed version of the raw data). On the other side, we also have quadratic and cubic algorithms such as hierarchical clustering [Duda et al., 2012] and spectral clustering [Ng et al., 2001, Zelnik-Manor and Perona, 2004, Von Luxburg, 2007] that use pairwise similarities to emphasize the latent clustering

structure lying on the data. Now, we describe some statistical problems related to the clustering task, and we will enumerate some famous clustering algorithms.

6.1.1 Clustering is an ill-posed problem

In general, there exists no clear, objective means of defining a "good clustering". For a fixed number of groups, Kleinberg [2015] presents three desirable properties for a given clustering algorithm, namely:

Scale Invariance Clustering output should not change if we multiply data by a constant

Richness or Cluster Shapes Invariance all separable cluster shapes should be possible (e. g. beyond linear separation or ball-shaped clusters)

Consistency or Metric Invariance Clustering output should not change with respect to the choice of distance

and he prooved the impossible existence of such an algorithm featuring all these three properties simultaneously. In other words, his clustering impossibility theorem tells us that clustering is an ill-posed problem. To add insult to injury, when data representation (or embedding) is involved, that clustering task becomes all the more unclear because the underlying metric is allowed to change arbitrarily. Indeed, the algorithm could expand the distance between points in the embedding space that initially were located near to each another, which would break initial pairwise "distance" constraints between the initial points and would inevitably violate the internal structure relating data. Well aware about these difficulties, we decided to try anyway following our scientific predecessors as clustering is useful in practice "as is".

Taking advantage of the abundant optimal transport literature with probabilities (relaxing hard clusters memberships definition to prefer probabilities) and also the neural networks literature (which successfully handles arbitrary classes shapes in supervised contexts) make our efforts reasonable towards a useful clustering algorithm for practioners. Let's review the clustering axioms of Kleinberg [2015]. First, once a metric is chosen, *scale-invariance* can be given for free thanks to the geometric optimal transport interpretations i. e. all 1-Wasserstein distance would be multiplied like the data accordingly without changing the optimization results. Second, Wasserstein distances operate on *all* pairs of distributions to the contrary of the Kullback-Leibler divergence (which requires common support which explains the use of infinite support distributions for the models like the Gaussian), and thus no cluster shapes assumption is required, thus *richness* would be fulfilled by the functional expressivity power provided by neural networks. Third, unfortunately, we would not be able to achieve *consistency* because our models and algorithms strongly depend on the euclidean distance. In fact, this third *consistency* property could be partially reached thanks to a generalized notion of Wasserstein distance defined by the maximum Wasserstein distance when the metric parses a family of distances (which makes the maximum of them still a distance) but this would require further scientific work that we just skimmed in our unsupervised feature importance contribution.

Richness (i. e. free clusters shapes robustness) is obtained thanks to an intermediate space that we call *embedding space* or *code space* which has lower dimensionality that the data space. If theoretical or practical tools are given to navigate between these spaces, back and forth without loosing too much information between data and codes thanks to encoder and decoder functions, then, model-based distribution assumptions can be made on the code side which gives us the richness property in return on the data side like in the work of Jiang et al. [2016]. Indeed, at the beginning of this thesis (mid-2016), our first intuition was to put a mixture distribution (which is a typical model-based idea) at the bottleneck of an auto-encoder (which is a typical deep learning unsupervised tool) with the hope of gathering the best of these two universes: probabilistic ease for model selection coming from model-based legacy on one side and rich representations with neural networks coming from deep learning legacy on the other side.

Fig. 10 represents our strategy to alleviate dimensionality issues while simulataneously performing clustering in a symbiotic fashion: this strategy proved successful in several papers that we briefly present here. The first work we saw doing clustering in the code space of an auto-encoder is the one

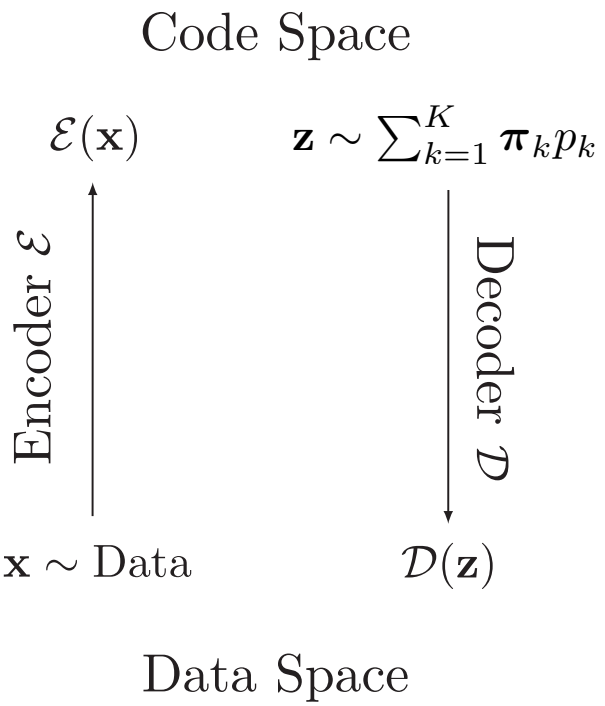

Figure 10: Data and codes spaces

of Song et al. [2014] and Yang et al. [2016a]. The idea is to cope with large data space dimensionality $D \gg 1$ thanks to an intermediate code space of lower dimensionality $d \ll D$. More precisely, Song et al. [2014] considered a k-Means-regularized auto-encoder loss to get a code space that is more easily clustered with k-Means namely their loss is the sum of the reconstruction and the k-Means residual with a chosen hyper-parameter combining both. This philosophy is the one adopted for our own approach but with a mixture of distributions $\sum_{k=1}^{K} \boldsymbol{\pi}_k \times p_k$ of distributions p_k weighted by proportions $\boldsymbol{\pi}$. We use a code space where clusters memberships (not data themselves but encoded versions of them) are easily computed. In our preliminary experiments, we found that optimizing the k-Means objective (online) when doing joint clustering and feature learning did not work well. We believe this is because it creates high magnitude gradients for points that are far away from cluster centers. Moreover there are sharp discontinuities at cluster boundaries whereas GMM diminishes that effect thanks to low density/probability values for far points. This empirical conclusion seems to confirm what Xie et al. [2015] also observed.

In a similar spirit, [Huang et al., 2014] have developped a locality-preserving and group-sparsity constraints method to handle the clustering. Yang et al. [2016b] alternate between supervised classification and feature learning through Convolutional neural networks (CNN) for images clustering. They significantly improved the state-of-the-art but their method is limited by its intrinsically quadratic complexity. In a similar spirit, Xie et al. [2015] embrace the t-SNE framework [Maaten and Hinton, 2008] in a clustering context through an auto-encoder in a non-model-based fashion. But doing t-SNE first and then clustering is not a good idea because of the same loss of useful cluster-wise information that occurred with PCA or auto-encoders.

All these works tend to show that simultaneous representation learning (by dimensionality reduction for example) and clustering actually do help each other. The reader will find an excellent review in the work of Aljalbout et al. [2018].

6.1.2 Clustering in Large-Scale Cardinality Regimes

k-Means and Mixture Models have been studied in large scale cardinality settings [Bottou and Bengio, 1995, Cappé and Moulines, 2009] but these algorithms work thanks to strong distribution assumptions (like cluster-wise Gaussian clusters shapes for GMM) directly in the original data space (*i.e.* without embeddings). Agglomerative clustering methods greedily use a square similarity matrix to fusion

data into clusters but the building of that $N \times N$ matrix is undoable for large cardinality N. Spectral clustering [Zelnik-Manor and Perona, 2004] works with very mild (or no) cluster shapes assumptions thanks to the kernel trick which gives access to high (and even infinite) dimensional representations space at the cost of a square similarity matrix once again which inevitably blocks the way leading to large cardinality datasets. Nevertheless, Choromanska et al. [2013] found a way to gracefuly alleviate this problem through to the the Nyström method that only demands the computations of only few entries of that non-storable square similarity matrix thanks to a low-rank approximation (which is justified by the low intrinsic manifold dimensionality hypothesis related to our *2D crinkled sheet of paper in a 3D room* metaphor). The present work is an attempt to provide a scalable method with the mildest possible cluster shapes assumptions thanks to neural networks that already have these two desirable properties in the supervised context: scalability and mild data assumptions.

The universal approximation theorems [Hanin and Sellke, 2017] allow neural networks to achieve richness (in the sense of Kleinberg [2015]) in theory. But in practice, that richness requires the estimation of a lot of parameters which is not reliable unless we have a large cardinality dataset during training compared to the dimensionality as explained above in section 5. In this case, large cardinality datasets are handled thanks to stochastic gradient optimization [Bach, 2016]. Indeed, considerable research in supervised classification has been conducted based on these foundations during the last decades but this work's challenge is about extending this success to unsupervised classification (a. k. a. clustering).

Generative approaches produce a model in the form of a synthetic data distribution that is supposed to be close to the original data distribution with respect to a criterion such as the Kullback-Leibler divergence (which is equivalent to maximizing the likelihood as explained by the *Pattern Recognition and Machine Learning* textbook of Bishop [2006]) typically optimized with Expectation-Maximization [Dempster et al., 1977]. Parameters and hyper-parameters are two different things: parameters are optimized whereas hyper-parameters are imposed before optimization and can be selected after optimization among a set of optimized models (i. e. model selection). One considerable advantage of generative techniques over others is that hyper-parameter selection is made easy through model selection thanks to principled mathematical (often Bayesian) foundations. Inddeed, building such a model for clustering gives strong tools to evaluate generalization capabilities (with famous criteria such as Akaike Information Criterion AIC, Bayesian Information Criterion BIC summarized by Duda et al. [2012] or even Integrated Completed Likelihood ICL [Biernacki et al., 2000] etc.).

Discriminative methods for clustering were initially inherited from supervised classification these last two or three decades. They are also extended to unsupervised classification (a. k. a. clustering). In clustering these discriminative approaches would not build a model that would fit the data but would rather separate the ouput classes or groups from each other (*e.g.* in a one-vs-one or one-vs-rest manner) focusing on the boundaries of the groups rather than on the groups themselves. Spectral Clustering [Von Luxburg, 2007] or DIFFRAC [Bach and Harchaoui, 2008] are two examples of such techniques.

6.1.3 k-Means solves an (Optimal) Transport Problem

We take a close look at the k-Means loss for data $(\mathbf{x}_i)_{i=1\dots N}$ into K groups:

$$\min_{\sigma,\boldsymbol{\mu}} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i - \boldsymbol{\mu}_{\sigma(i)}\|_2^2 \tag{24}$$

that we optimize over the centroids $\boldsymbol{\mu} = (\boldsymbol{\mu}_k)_{k=1\dots K}$ and the assignments function σ from $[\![1, N]\!] \subset \mathbb{N}$ to $[\![1, K]\!] \subset \mathbb{N}$.

We studied k-Means which is probably both the oldest and most famous clustering algorithm [Jain, 2010] and we realized that it tries to efficiently solve an optimal transport problem. Put differently, the global minimum of the k-Means loss satisfies the optimal transport problem of choosing a limited K number centroids $(\boldsymbol{\mu}_k)_{k=1,\dots,K}$ such that the data empirical distribution $p = \frac{1}{N}\sum_{i=1}^{N} \delta_P \mathbf{x}_i$ on the one hand and the centroids weighted distribution $q = \sum_{k=1}^{K} \boldsymbol{\pi}_k \times \delta_{\boldsymbol{\mu}_k}$ on the other hand would be the closest possible in the 2-Wasserstein sense associated to the squared euclidean distance, although

usually, we use the 1-Wasserstein distance associated with the plain and simple euclidean distance:

$$W_{c_2}(p,q) = \min_{\gamma \in \Gamma(p,q)} \mathbb{E}_{(\mathbf{x},\mathbf{m})\sim\gamma} \left[\|\mathbf{x}-\mathbf{m}\|_2^2 \right] \tag{25}$$

and for discrete distributions:

$$W_{c_2}(p,q) = \frac{1}{N} \sum_{k=1}^{K} \boldsymbol{\pi}_k \times \mathbb{E}_{\mathbf{x}\sim p} \left[\|\mathbf{x}-\boldsymbol{\mu}_k\|_2^2 \right] \tag{26}$$

which is the k-Means loss and of course we have a link between the proportions $\boldsymbol{\pi}$ and the assignments $\boldsymbol{\sigma}$: $\boldsymbol{\pi}_k = \frac{\#\{\boldsymbol{\sigma}(i)=k|\, i\in[\![1,N]\!]\}}{N}$ as for discrete distributions, optimal transport plans are degenerated [Peyré et al., 2019].

This interesting link between clustering and optimal transport encouraged us to investigate further between these two scientific literatures: clustering and optimal transport. Generalizing this observation to more sophisticated distributions thanks to Generative Adversarial Networks could lead to having a *fatter* support distribution than just Diracs:

$$\min_{\boldsymbol{\sigma},(\boldsymbol{\theta}_{p_k})_{k=1\ldots K}} W(\sum_{k=1}^{K} \boldsymbol{\pi}_k \times p_k, \frac{1}{N} \sum_{i=1}^{N} \times \delta_{\mathbf{x}_i}) \tag{27}$$

where $\boldsymbol{\theta}_{\mathbf{p}_k}$ parametrized the kth cluster generator distribution $\mathbf{p}_k$ (that was previously reduced to a Dirac distribution located on the kth centroid $\boldsymbol{\mu}_k$). In practice, we can have latent variables $\mathbf{z} \sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$ and we define in $\mathbf{y}_k \sim p_k$ by $\mathbf{y} = \mathcal{G}_k(\mathbf{z})$ as we will develop later. That initial idea we had is encouraging in a sense that it suggests an interesting mixing between optimal transport, neural networks and model-based clustering.

We think that there is an opportunity here to mention that k-Means is *not* a regular special case of Gaussian Mixture Models through Expectation Maximization (EM-GMM) although this false idea is widespread. It is true that when neg-exponentiated, the k-Means looks like the negative likelihood of a Gaussian mixture fitted on the data with identity (or equally proportional to) covariance matrices and equal proportions but there is major difference: memberships probabilities in EM-GMM are not degenerated but they are in k-Means. k-Means can still be seen as an example of the Expectation-Maximization technique but these distinctions have been clearly made by Celeux and Govaert [1992]. The link between k-Means and optimal transport is stronger than the one between k-Means and EM-GMM and also more fruitful in terms of open research.

6.2 Unsupervised Feature Importance

As suggested earlier when we analyzed the consequences of the impossibility theorem by Kleinberg [2015], metric invariance is an interesting subject. To handle this difficulty related to the choice of the metric, one can work with a set of metrics because we know that the metric defined by the upper bound evaluation over a set of ditances is also a distance. To the best of our knowledge, this angle has not been tackled by the research community to study unsupervised feature importance extraction.

In supervised learning, Breiman [2001] proposed routines based on permutation and mean decrease in impurity but much less work has been done in the unsupervised context. This is probably due to the fact two interleaved problems remain: metric learning, feature selection which makes our task ill-posed. We actually suggest that it is worth trying to improve regular and generic euclidean approaches.

6.3 Uncertain Predictions

In this chapter devoted to uncertainty in supervised learning, we borrow scientific items from Bayesian and frequentist scientific communities. Indeed, we believe that mixing both scientific cultures is beneficial in general and for uncertainty estimation in particular: probabilistic interpretations provided by Bayesian formulations and function expressivity fitted to large scale data provided by the large frequentist fauna of algorithms which is of course not limited to deep neural networks.

6.3.1 Bayesian and Frequentist scientists in Statistical Learning

There is a so-called rivalry in automatic statistical learning between Bayesians on one side and frequentists on the other side that is quiet disturbing when seen from a young scientific point of view. Indeed, we can laughably notice that Bayesian scientists are allowed to use histograms of frequencies and frequentist scientists are allowed to use the Bayes rule. This kind of debate is often sterile but it is probably a characteristic of still young non-unified sciences with varying names accross trends. In this section, we briefly describe the specific issues taken into account by these two commonly separated communities.

The Bayesian framework [Barber, 2011] is characterized by its use of parametrized probability laws which allows to benefit from interpretation ease (including uncertainty) when predicting information of producing description about data. The preferred statistical tool is usually the so-called bayes rule, hence the name of this scientific community. The choice of the modelled distributions carries interpretation and knowledge that is elegantly injected into the trained systems.

In contrast, within the frequentist approach [Bishop, 2006], we assume that uncertainty is inherenty present due to the randomness coming from repeatable experiments producing empirical observations. Hence, many machine learning problems tackled with a frequentist point of view is a statistical estimation problem based on observed data. If one could accept caricatures, then we would say that Bayesian statistical learning is a principled probabilistic and thus interpretable framework (hence their appealing reputation but at the cost of often wrong model assumptions) which offers extraordinary research avenues such as model selection without extra data and meaningful probability interpretations whereas frequentist statistical learning produce good-results-oriented black-boxes [Neal, 1995] with impressive recognition rates resculpting our modern world.

Recently, in a Ph.D. thesis Gal [2016], dared the idea of taking advantage of both worlds in a Bayesian deep learning approach (although the merit was certainly to revisit such a counter-intuitive approach that in fact dates back to at least in the 1990s by Neal [1995] or even by Bishop [1994]). Today in 2020, there is a still a controversial debate on this subject: Bayesian machine learning injects some knowledge through a distribution prior (not always a Gaussian prior even if we can recognize this is the best studied distribution and the most frequently used) for inputs and outputs of statistical predictors. This guiding of the machine learning at both optimization and prediction steps assumes some knowledge about input and output data but most of the time that knowledge is not existing hence the debate. Frequentist neural networks with all their parameters and Bayesian statistics with all their parametrized distributions might seem uneasy to coexist in the same unique machine learning method.

6.3.2 Sources of Uncertainty

Beyond frequentist and Bayesian considerations, according to recent research work [Gal, 2016] (we recommend the reader to read this Ph.D. thesis for details and comprehensive bibliography about prediction with uncertainty), uncertainty can be broken down into several facets:

Extrapolation uncertainty The prediction could be wrong because test data do not come from the same distribution as training data. Out from the training distribution, test data go against one of the most fundamental machine learning hypothesis to make any system work. For example, online self-adapting systems [Bertsekas et al., 1995] look like reinforcement learning systems and deal with out of distribution data uncertainty purposefully. During training, prediction systems never see anything but data coming from training data. Neural networks have the deserved reputation of quickly over-fitting on training data empirical distribution (if used carelessly) which is both a warning against both interpolation and extrapolation data prediction in worst case scenarios as explained by **?** partially avoided by regularization [Srivastava, 2013]. Overfitting is bad extrapolation in essence. Thus, neural networks are particularly prone to extrapolation issues like extrapolation uncertainty.

Aleatoric uncertainty Some noise could have been introduced in training data (some wrong coordinates, some wrong labels etc.). Well-studied statistical machineries like linear and kernel-based

support vector machines [Andrew, 2001] proposed the notion of margin to cope with some part of that uncertainty but it seems that linking that margin to probability estimation is still an open research problem even with a logistic regression loss instead of a hyperplane in practice;

Epistemic uncertainty The initial machine learning problem might be ill-defined: many solutions can solve the problem, so we do not know which one to choose objectively. For example, the *butterfly effect* (in layman's terms) is a source of epistemic uncertainty against meteorological forecasts meaning that uncertainty is inherently related to the studied laws of physics as a scientific standing point (see for example Epstein [1969]). Thus, the problem modeling could be unsufficient which introduces randomness.

There exists more precise analysis for describing these interleaved sources of uncertainty and readers may find more exhaustive research in the work of Kennedy and O'Hagan [2001]. But all these phenomena boil down to how systems should and could handle the possible lack of confidence tainting automatic predictions.

References

S. Abbott and C. A. Rogers. *Hausdorff Measures*, volume 83. Cambridge University Press, 1999. doi: 10.2307/3619107.

A. Abouchar. Air Transport Demand, Congestion Costs, and the Theory of Optimal Airport Use. *The Canadian Journal of Economics*, 3(3):463, 1970. ISSN 00084085. doi: 10.2307/133661.

S. Abu-El-Haija, N. Kothari, J. Lee, A. P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. YouTube-8M: A Large-Scale Video Classification Benchmark. *Google Research*, 2016. URL `https://research.google/pubs/pub45619`.

Z. Ahmad. *The epistemology of Ibn Khaldūn*. Routledge, 2003. ISBN 020363389X.

R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. Network flows. *Handbooks in Operations Research and Management Science*, 1(C):211–369, 1989. ISSN 09270507. doi: 10.1016/S0927-0507(89)01005-4.

G. Alain and Y. Bengio. What regularized auto-encoders learn from the data-generating distribution. *Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

E. Aljalbout, V. Golkov, Y. Siddiqui, and D. Cremers. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*, 2018.

A. M. Andrew. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, volume 30. Cambridge University Press, 2001. doi: 10.1108/k.2001.30.1.103.6.

C. Archambeau and M. Verleysen. Manifold constrained variational mixtures. In *International Conference on Artificial Neural Networks*, pages 279–284. Springer, 2005.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2017. URL `http://proceedings.mlr.press/v70/arjovsky17a.html`.

S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *34th International Conference on Machine Learning, ICML 2017*, volume 1, pages 322–349. JMLR, 2017. ISBN 9781510855144.

F. Bach. Beyond stochastic gradient descent for large-scale machine learning, 2016. URL `http://ecmlpkdd2014.loria.fr/wp-content/uploads/2014/09/fbach{_}ecml{_}2014.pdf`.

F. R. Bach and Z. Harchaoui. Diffrac: a discriminative and flexible framework for clustering. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 49–56. Curran Associates, Inc., 2008. URL `http://papers.nips.cc/paper/3269-diffrac-a-discriminative-and-flexible-framework-for-clustering.pdf`.

P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2011. doi: 10.1017/cbo9780511804779.

E. Barillot, L. Calzone, P. Hupe, J.-P. Vert, and A. Zinovyev. *Computational systems biology of cancer*. CRC Press, 2012.

R. E. Bellman. *Dynamic Programming*. Rand Corporation Research Study. Princeton University Press, 1957.

J. D. Benamou and Y. Brenier. A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000. ISSN 0029599X. doi: 10.1007/s002110050002.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. ISSN 01628828. doi: 10.1109/TPAMI.2013.50.

C. Bernard. *Introduction à l'étude de la médecine expérimentale*. Librairie Joseph Gilbert, 1898.

D. P. Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997. ISSN 01605682. doi: 10.4018/978-1-4666-5202-6.ch147.

D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas. *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995.

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000. doi: 10.1109/34.865189.

C. Bishop. Mixture density networks. Technical report, January 1994.

C. M. Bishop. Pattern recognition. *Machine Learning*, 128, 2006.

P. Bojanowski and A. Joulin. Unsupervised learning by predicting noise. *arXiv preprint arXiv:1704.05310*, 2017.

M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars. *Thirtieth Conference on Neural Information Processing Systems*, abs/1604.07316, 2016. URL `https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf`.

J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal. *Numerical optimization: theoretical and practical aspects*, volume 41. Springer Science and Business Media, 2003. doi: 10.5860/choice.41-0357.

L. Bottou and Y. Bengio. Convergence Properties of the K-Means Algorithms. In *Advances in Neural Information Processing Systems*, pages 585–592, 1995.

H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

C. Bouveyron and C. Brunet. Simultaneous model-based clustering and visualization in the fisher discriminative subspace. *Statistics and Computing*, 22(1):301–324, 2012.

C. Bouveyron and C. Brunet-Saumard. Model-based clustering of high-dimensional data: A review. *Computational Statistics and Data Analysis*, 71:52–78, 2014.

C. Bouveyron, S. Girard, and C. Schmid. High-Dimensional Data Clustering. *Computational Statistics and Data Analysis*, 2007.

C. Bouveyron, G. Celeux, T. B. Murphy, and A. E. Raftery. *Model-Based Clustering and Classification for Data Science: With Applications in R*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014. ISBN 978-0-521-83378-3. doi: 10.1017/CBO9780511804441. URL `https://web.stanford.edu/%7Eboyd/cvxbook/`.

L. Breiman. Probability, volume 7 of classics in applied mathematics. *Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA*, 2:6, 1992.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, oct 2001. ISSN 08856125.

L. Breiman. *Classification and regression trees*. Routledge, 2017. ISBN 9781351460491.

A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. *7th International Conference on Learning Representations, ICLR 2019*, 2019.

O. Cappé and E. Moulines. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, 2009.

M. Castelluccio. AI rising. *Strategic Finance*, 2017.

G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14(3):315–332, 1992. ISSN 01679473. doi: 10.1016/0167-9473(92)90042-E. URL `http://www.sciencedirect.com/science/article/pii/016794739290042E`.

A. Chakraborty, S. Ghosh, P. Mukhopadhyay, S. M. Dinara, A. Bag, M. K. Mahata, R. Kumar, S. Das, J. Sanjay, S. Majumdar, and D. Biswas. Trapping effect analysis of AlGaN/InGaN/GaN Heterostructure by conductance frequency measurement. *MRS Proceedings*, XXXIII(2):81–87, 2014. ISSN 0717-6163. doi: 10.1007/s13398-014-0173-7.2.

T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, 24(12):5017–5032, 2015.

W.-C. Chang. On using principal components before separating a mixture of two multivariate normal distributions. *Applied Statistics*, pages 267–275, 1983.

G. Chen. Deep learning with nonparametric clustering. *arXiv preprint arXiv:1501.03084*, 2015.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 1724–1734, 2014. doi: 10.3115/v1/d14-1179.

A. Choromanska, T. Jebara, H. Kim, M. Mohan, and C. Monteleoni. Fast spectral clustering via the nyström method. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 367–381, 2013. ISBN 9783642409349. doi: 10.1007/978-3-642-40935-6_26.

N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal Transport for Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1853–1865, 2017.

N. Crasto, P. Weinzaepfel, K. Alahari, and C. Schmid. MARS: Motion-Augmented RGB Stream for Action Recognition. In *CVPR*, 2019.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2292–2300. 2013.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani. *Algorithms*. McGraw-Hill Higher Education, 2008.

F. De la Torre and T. Kanade. Discriminative cluster analysis. pages 241–248, 2006.

H. De March. *Multidimensional martingale optimal transport.* Theses, Université Paris-Saclay, June 2018. URL `https://pastel.archives-ouvertes.fr/tel-01973279`.

A. Defazio, F. R. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 1646–1654, 2014.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real nvp. *International Conference on Learning Representations*, 2017.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley and Sons, 2012.

R. M. Dudley. *Real analysis and probability*. Chapman and Hall/CRC, 2018.

V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. In *International Conference on Learning Representations*, 2017.

E. S. Epstein. Stochastic dynamic prediction. *Tellus*, 21(6):739–759, 1969. doi: 10.3402/tellusa.v21i6.10143.

W. Fedus, M. Rosca, B. Lakshminarayanan, A. M. Dai, S. Mohamed, and I. Goodfellow. Many paths to equilibrium: Gans do not need to decrease a divergence at every step. *arXiv preprint arXiv:1710.08446*, 2017.

L. Fei-Fei. Imagenet: crowdsourcing, benchmarking and other cool things. In *CMU VASC Seminar*, volume 16, pages 18–25, 2010.

Y. Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

J. K. Galbraith. The pragmatism of John Kenneth Galbraith. *Acta Oeconomica*, 69(s1):195–213, 2019. ISSN 15882659. doi: 10.1556/032.2019.69.S1.12.

T. Gao and V. Jojic. Degrees of freedom in deep neural networks. *32nd Conference on Uncertainty in Artificial Intelligence 2016, UAI 2016*, 2016.

A. Genevay. *Entropy-regularized optimal transport for machine learning*. PhD thesis, 2019.

T. Glasmachers. Limits of end-to-end learning. *arXiv preprint arXiv:1704.08305*, 2017.

A. V. Goldberg and R. E. Tarjan. Finding Minimum-Cost Circulations by Canceling Negative Cycles. *Journal of the ACM (JACM)*, 36(4):873–886, 1989. ISSN 1557735X. doi: 10.1145/76359.76368.

Y. Goldberg. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309, 2017.

G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

I. J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. 2016. URL `http://arxiv.org/abs/1701.00160`.

B. Hanin and M. Sellke. Approximating continuous functions by relu nets of minimal width. *CoRR*, abs/1710.11278, 2017. URL `http://arxiv.org/abs/1710.11278`.

Z. Harchaoui. Large-scale learning for image classification, 2013. URL https://harchaoui.org/zaid/cvml13.pdf.

T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

P. J. Hayes and S. P. Weinstein. Construe-TIS: A System for Content-based Indexing of a Database of News Stories. In *Second Annual Conference on Innovative Applications of Artificial Intelligence*, volume 90, pages 49–64, 1990. ISBN 0-262-68068-8.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. URL https://arxiv.org/abs/1512.03385.

H. Hendrikx, F. Bach, and L. Massoulié. Accelerated decentralized optimization with local updates for smooth and strongly convex objectives. In K. Chaudhuri and M. Sugiyama, editors, *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 897–906. PMLR, 2019. URL http://proceedings.mlr.press/v89/hendrikx19a.html.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2): 251–257, 1991.

P. Huang, Y. Huang, W. Wang, and L. Wang. Deep embedding network for clustering. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 1532–1537. IEEE, 2014.

A. K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 2010. ISSN 01678655.

R. Jenatton. *Structured sparsity-inducing norms: Statistical and algorithmic properties with applications to neuroimaging*. PhD thesis, 2011.

Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. Variational deep embedding: A generative approach to clustering. 2016. URL http://arxiv.org/abs/1611.05148.

M. I. Jordan. On statistics, computation and scalability. *Bernoulli Society for Mathematical Statistics and Probability*, abs/1309.7804, 2013. URL http://arxiv.org/abs/1309.7804.

O. Kallenberg. *Random measures, theory and applications*, volume 77. Springer, 2017.

A. Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks, 2015. URL http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001. ISSN 1369-7412. doi: 10.1111/1467-9868.00294.

J. M. Kleinberg. An impossibility theorem for clustering. In *Advances in Neural Information Processing Systems*, pages 463–470, 2015.

M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2 (1-2):83–97, 1955.

Y. LeCun. What's Wrong With Deep Learning? In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. URL `http://yann.lecun.com`.

Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *MIT Press, Cambridge*, 1995.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, 1989a. ISSN 0899-7667.

Y. LeCun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard. Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning. *IEEE Communications Magazine*, 27(11):41–46, 1989b.

Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems 2, NIPS 1989*, pages 396–404. Morgan Kaufmann Publishers, 1990.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 1998.

B. Lévy. A Numerical Algorithm for L2 Semi-Discrete Optimal Transport in 3D. *ESAIM: Mathematical Modelling and Numerical Analysis*, 49(6):1693–1715, 2015. ISSN 12903841. doi: 10.1051/m2an/2015055.

J. Li, L. Deng, R. Haeb-Umbach, and Y. Gong. *Fundamentals of speech recognition*. Pearson Education India, 2016. doi: 10.1016/b978-0-12-802398-3.00002-7.

Y. Liu, M. Yamada, Y. H. Tsai, T. Le, R. Salakhutdinov, and Y. Yang. Lsmi-sinkhorn: Semi-supervised squared-loss mutual information estimation with optimal transport. *AAAI*, abs/1909.02373, 2020. URL `http://arxiv.org/abs/1909.02373`.

L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9 (Nov):2579–2605, 2008.

J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on image processing*, 17(1):53–69, 2008.

A. Makhzani, J. Shlens, N. Jaitly, and I. Goodfellow. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

S. Mallat. *A wavelet tour of signal processing: the sparse way*. Academic Press, Inc., 2008.

V. G. Maz'ya and T. O. Shaposhnikova. *Jacques Hadamard: a universal mathematician*. Number 14. American Mathematical Soc., 1999.

Q. Mérigot. A multiscale approach to optimal transport. In *Eurographics Symposium on Geometry Processing*, volume 30, pages 1583–1592. Wiley Online Library, 2011. doi: 10.1111/j.1467-8659.2011.02032.x.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations ofwords and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

G. Monge. Mémoire sur la théorie de déblais et de remblais. Histoire de l'Académie Royale des science de Paris, avec les Mémoires de Mathématiques et de Physique pour la même année, 1781. URL https://gallica.bnf.fr/ark:/12148/bpt6k35800/f1.image.

K. Murakami and H. Taguchi. Gesture recognition using recurrent neural networks. In *Conference on Human Factors in Computing Systems - Proceedings*, pages 237–242. ACM, 1991. ISBN 0897913833. doi: 10.1145/108844.108900.

K. Murphy. *Machine Learning, a Probabilistic Perspective*. MIT press, 2012.

R. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

S. Newman. *Descartes' epistemology*. Routledge, 2018.

A. Ng. Deep learning, 2013. URL https://www.youtube.com/watch?v=n1ViNeWhC24.

A. Ng. The state of artificial intelligence, 2018. URL https://www.youtube.com/watch?v=19IXayufFv4.

A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. 14(2):849–856, 2001.

K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

B. Peters. The Age of Big Data. *Forbes*, 11(2012):4–9, 2012. URL http://www.forbes.com/sites/bradpeters/2012/07/12/the-age-of-big-data/.

G. Peyré, M. Cuturi, et al. Computational optimal transport. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

J. Ponce and D. Forsyth. *Computer vision: a modern approach*. 2011.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

J. Rabin, G. Peyré, J. Delon, and M. Bernot. Wasserstein barycenter and its application to texture mixing. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 435–446. Springer, 2012.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

L. Roberts. Picture Coding Using Pseudo-Random Noise. *IRE Transactions on Information Theory*, (2): 145–154, 1962. ISSN 21682712.

F. Rosenbaltt. The perceptron–a perciving and recognizing automation. *Report 85-460-1 Cornell Aeronautical Laboratory, Ithaca, Tech. Rep.*, 1957.

F. Rosenblatt. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Technical Report 4, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

F. Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

C. Schmid. Active Large-scale Learning for Visual Recognition, 2013. URL https://lear.inrialpes.fr/allegro.

B. Scholkopf and A. J. Smola. *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2001.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

J. Shawe-Taylor, N. Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

D. Shu, J. Cunningham, G. Stump, S. W. Miller, M. A. Yukish, T. W. Simpson, and C. S. Tucker. 3D Design Using Generative Adversarial Networks and Physics-based Validation. *Journal of Mechanical Design*, 142(7):1–51, 2019. ISSN 1050-0472. doi: 10.1115/1.4045419.

K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The Hadoop distributed file system. In *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies, MSST2010*, 2010.

L. Smolinski. The Scale of Soviet Industrial Establishments. *The American Economic Review*, 52(2): 138–148, 1962. URL `https://about.jstor.org/terms`.

R. Socher, C. C. Lin, A. Y. Ng, and C. D. Manning. Parsing natural scenes and natural language with recursive neural networks. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML*, pages 129–136. Omnipress, 2011. ISBN 9781450306195. URL `https://icml.cc/2011/papers/125_icmlpaper.pdf`.

C. Song, Y. Huang, F. Liu, Z. Wang, and L. Wang. Deep auto-encoder based clustering. *Intelligent Data Analysis*, 18(6S):S65–S76, 2014.

S. Sonoda and N. Murata. Decoding stacked denoising autoencoders. *arXiv preprint arXiv:1605.02832*, 2016.

N. Srivastava. Improving neural networks with dropout. *University of Toronto*, 182(566):7, 2013.

K. Sung, T. Poggio, H. Rowley, S. Baluja, and T. Kanade. MIT+ CMU frontal face dataset a, b and c, 1998.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June-2015, pages 1–9, 2015. ISBN 9781467369640.

J. Thickstun, Z. Harchaoui, D. P. Foster, and S. M. Kakade. Invariances and Data Augmentation for Supervised Music Transcription. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018.

I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=HkL7n1-0b`.

G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller. A deep semi-nmf model for learning hidden representations. In *ICML*, pages 1692–1700, 2014.

M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 586–591. IEEE, 1991.

J.-P. Vert. Learning from ranks, learning to rank. URL `http://members.cbio.mines-paristech.fr/~jvert/talks/200114turing/turing.pdf`.

R. Vidal, J. Bruna, R. Giryes, and S. Soatto. Mathematics of deep learning, 2017. URL `https://www.youtube.com/watch?v=eEPXTMHnBJA`.

C. Villani. *Optimal transport: old and new*, volume 338. Springer Science and Business Media, 2008.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

P. A. Viola and M. J. Jones. Robust real-time face detection. In *Proceedings of the Eighth International Conference On Computer Vision (ICCV-01), Vancouver, British Columbia, Canada, July 7-14, 2001 - Volume 2*, page 747. IEEE Computer Society, 2001.

U. Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Z. Wang, S. Chang, J. Zhou, M. Wang, and T. S. Huang. Learning a task-specific deep architecture for clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 369–377. SIAM, 2016.

J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. *arXiv preprint arXiv:1511.06335*, 2015.

B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. *arXiv preprint arXiv:1610.04794*, 2016a.

J. Yang, D. Parikh, and D. Batra. Joint Unsupervised Learning of Deep Representations and Image Clusters. 2016b. URL `http://arxiv.org/abs/1604.03628`.

M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. *2nd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud 2010*, 2010.

M. Zaslavskiy. *Graph matching and its application in computer vision and computational biology R ´ esum ´ e*. PhD thesis, Mines de Paris, 2010.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. *Advances in neural information processing systems*, 17(1601-1608):16, 2004.

X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657, 2015.

P. Zikoupoulos and C. Eaton. *Understanding big data: Analytics for Enterprise Class Hadoop and Streaming*, volume 11. McGraw-Hill Osborne Media, 2016. ISBN 9780071790536.

V. Zue, S. Seneff, and J. Glass. Speech database development at MIT: Timit and beyond. *Speech Communication*, 1990. ISSN 01676393.