SAPIENZA
UNIVERSITÀ DI ROMA

Improving Reliability in Deep Learning: Exploring Generalization Bounds, Noisy Label Handling, and Loss Surface Characterization

Faculty of Information Engineering, Informatics, and Statistics
Department of Computer, Control, and Management Engineering "Antonio Ruberti"

PhD in Data Science

Maria Sofia Bucarelli
1617005

Thesis Advisor
Fabrizio Silvestri

Academic Year 2022/2023 (XXXVI cycle)

Improving Reliability in Deep Learning: Exploring Generalization Bounds, Noisy Label Handling, and Loss Surface Characterization
Sapienza University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: mariasofia.bucarelli@uniroma1.it

Abstract

This dissertation investigates the theory of generalization and robustness in deep learning. Through diverse research works, the thesis provides valuable insights and advancements towards building more reliable systems. We focus on handling noisy labels, deriving generalization bounds for clustering, enhancing interpretability, and characterizing the topology of the loss landscape. These findings contribute to the broader field of deep learning, advancing the development of effective and reliable machine learning systems. After the introduction provided in Chapter 1, in the second chapter we tackle the challenge of noisy labels in classification by leveraging inter-rater agreement and estimating the noise distribution, thereby improving model performance and robustness. In the third chapter, we establish generalization bounds for projective clustering, presenting near-optimal results for subspace clustering. Chapter 4 introduces a novel artificial neuron that enhances interpretability while retaining the representation power and performance of a standard neural network. Indeed, we prove the universal approximation theorem for specialized versions of the artificial neuron. In Chapter 5, we characterise the topological complexity loss surfaces using Betti Numbers. Understanding the topology of loss surfaces is crucial for studying generalization and robustness in deep learning models. The last chapter summarizes the key findings and contributions, also mentioning possible future directions. Collectively, this research work contributes to understanding generalization and robustness in deep learning, advancing the field and enabling the development of more reliable models.

Acknowledgments

The first acknowledgement goes to Fabrizio, my advisor. I am highly grateful for having him as my Ph.D. advisor, for his invaluable support, for his patience in guiding the group working with him, for inspiring the team with his passion, and for the infectious enthusiasm with which he leads our lab. I have learned so much in these three years. I am fortunate to have been part of an extraordinary team that works with Fabrizio. I express my heartfelt gratitude to each member for contributing to creating an outstanding group characterized by a pleasant atmosphere and exceptional competence.

During the Ph.D. I had the opportunity to visit two amazing groups: Pietro Liò in Cambridge and Chris Schwiegelshohn in Aarhus.

They have all been incredibly welcoming and supportive, and I learned so much!

Being part of the dynamic group led by Pietro in Cambridge was a highly valuable experience. I gleaned extensive insights from Pietro's broad vision on machine learning problems. The supportive environment and the lab's expertise allowed me to acquire significant knowledge and exposure by engaging with diverse research environments.

Additionally, I would like to express my gratitude to Chris for his invaluable guidance, which significantly contributed to my academic learning and progress.

My gratitude goes to all the fantastic people I met during these travels: Olga, Charlotte, Francesco, Francesco, Lorenzo, Donato, Pietro, Simon, Alisia, Amik, and many others.

For a shorter period of time, I also visited the group of Franco Scarselli and Monica Bianchini in Siena to work with Alessio D'Inverno. I'm grateful to them for inviting me to work in a warm and comfortable environment.

In the second year of my Ph.D., I had the great opportunity to join Amazon Science as an intern hosted by Amin Mantrach. I worked alongside a fantastic team of engineers and researchers. This exposure granted me a valuable applied view of research that has significantly enriched me. I want to express my sincere gratitude to every team member, especially my manager, Amin and my mentor, Lucas, for making my internship highly rewarding.

I am grateful to all my Ph.D. colleagues who have shared the lab with me, fostering companionship during numerous lunches and workdays.

I'd like to thank my former classmates in the Math department Federico, Francesco, and Andrea. Even though we pursued our doctoral programs in different departments and cities, their support and encouragement meant a lot to me throughout this journey.

I extend my deepest gratitude to my lifelong friends and family who have consistently been a part of my life outside the university. Their enduring love and affection have been an immeasurable source of support over these three years.

Contents

Origins of the thesis

Publications covered in this Thesis.

- ☐ Bucarelli, M. S., Cassano, L., Siciliano, F., Mantrach, A., and Silvestri, F. (2023a). Leveraging inter-rater agreement for classification in the presence of noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3439–3448

  MSB (Maria Sofia Bucarelli) designed the method, proved the theoretical results and assisted FS in designing the experiments. All authors contributed to the writing, MSB did most of the writing. This work was done during her internship at Amazon.

- ☐ Bucarelli, M. S., Larsen, F. M., Schwiegelshohn, C., and Toftrup, M. B. (2023b). On generalization bounds for projective clustering. In Oh, A., Nauman, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*

  CS, MSB and ML proved the theoretical results. All authors contributed to the writing.

- ☐ Siciliano, F., Bucarelli, M. S., Tolomei, G., and Silvestri, F. (2022). Newron: A new generalization of the artificial neuron to enhance the interpretability of neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 01–17

  MSB proved the theoretical results on the approximation properties of the models and significantly contributed to the paper's writing.

Additionally, the thesis contains some unpublished material, such as the study of topological surfaces of the loss via Betti numbers in Chapter 5, which is a research work conducted in collaboration with Giuseppe Alessio D'Inverno, Monica Bianchini and Franco Scarselli from the University of Siena.

Other works done during the Ph.D.

Wani, F. A., Bucarelli, M. S., and Silvestri, F. (2023). Combining distance to class centroids and outlier discounting for improved learning with noisy labels. *arXiv preprint arXiv:2303.09470*

Telyatnikov, L., Bucarelli, M. S., Bernardez, G., Zaghen, O., Scardapane, S., and Lio, P. (2023). Hypergraph neural networks through the lens of message passing: A common perspective to homophily and architecture design. *arXiv preprint arXiv:2310.07684*

Bragatto, T., Bucarelli, M. A., Bucarelli, M. S., Carere, F., Geri, A., and Maccioni, M. (2023). False data injection impact on high res power systems with centralized voltage regulation architecture. *Sensors*, 23(5):2557

Trappolini, G., Cassarà, G., Bucarelli, M. S., and Silvestri, F. (2023). A simple yet tough to beat baseline to counterfactually explain gnns. In *Submitted to AISTATS 2024*

Siciliano, F., Magister, L. C., Bucarelli, M. S., Barbiero, P., Silvestri, F., and Lio, P. (2023). Explaining neural networks using a ruleset based on interpretable concepts. In *Submitted to EPJ Data Science*

Chapter 1

Introduction

The central objective of this work is to better understand machine learning models and enhance their reliability. Reliability, in this context, refers to dependable, consistent, and trustworthy models. Indeed, we aim to create systems in which efficiency and results are always consistent and accurate over time and under different conditions. Providing theoretical guarantees to machine learning models is essential to strengthen their trustworthiness and reliability. It not only instils confidence in the model's predictions but also aids in model selection and serves as a sturdy cornerstone for the development and deployment of machine learning models.

Background

Machine Learning (ML) is focused on enabling systems to leverage data for learning relationships between variables. Much of machine learning concerns devising different models and algorithms to fit them (Murphy, 2012).

The "No Free Lunch" theorem emphasizes that, without prior knowledge, all algorithms are equally bad, on average, when considering all possible problems (Wolpert and Macready, 1997).

However, when we bring in domain-specific knowledge, we can narrow our search for a suitable algorithm or hypothesis set. The *hypothesis set* is the collection of potential candidate solutions that a machine learning algorithm considers when solving a specific problem. It is the set of all possible maps that we restrict our optimization algorithm to find the best-fit model for the given data. For instance, consider a linear regression scenario where the goal is to predict housing prices (y) based on the house size (x). The hypothesis set, $\mathcal{F}$, includes all possible linear models $y = mx + b$. Here, the optimization algorithm explores various combinations of slopes (m) and intercepts (b) to find the best-fitting line that describes the relationship between house size and price using the provided data.

In the usual scenario, we are given a set of training data $S = (Z(i))_{i=1}^{m}$, where Z_i are drawn from an unknown distribution $\mathbb{P}_Z$.

Generally, the effectiveness of a solution f is evaluated using an *objective function* $\mathcal{L}(f, Z)$ that we aim to optimize.

We denote by f^* the function that optimizes the objective $\mathcal{L}$, considering the data distribution $\mathbb{P}_Z$. Additionally, we denote by $f_{S,\mathcal{F}}$ the approximation of the solution achieved through our optimization process. This approximation is based on the constraints of using the hypothesis set $\mathcal{F}$ and the dataset S.

The error of the solution $f_{S,\mathcal{F}}$ with respect to the optimal solution f^* can be bound by three different errors: the approximation error, the generalization error and the optimization error (Berner et al., 2021).

The **approximation error** tells us how good the hypothesis set is; it measures how far the entire hypothesis class is from the target predictor f^*. Larger hypothesis classes have lower approximation errors.

The **optimization error**, instead, tells us how good the function returned by the learning algorithm is with respect to the best map in the hypothesis class. This error is primarily influenced by the numerical algorithm used to find the model in a hypothesis set. In classical theory, the optimization error increases monotonically as the hypothesis class size increases (Berner et al., 2021).

Finally, the **generalization error** arises because we are finding the best solution using only a given sample of points S without knowing their underlying distribution $\mathbb{P}_Z$. This error tells us how well the solution found using the m size set S generalizes to the distribution $\mathbb{P}_Z$. In this case, the most basic requirement one can ask for is that as $m \to \infty$, the generalization error goes to zero. The approximation error decreases when enlarging the hypothesis set, but considering all possible models prevents controlling the generalization error (Wolpert and Macready, 1997; Devroye et al., 2013).

Generalization is a crucial concept in machine learning, as the ultimate objective of training a machine learning model is not merely to memorize the training data but to learn and generalize from it in a way that allows the model to perform well on unseen data.

Generalization bounds provide a theoretical foundation for assessing the performance and reliability of machine learning models. They offer an upper limit on the expected error of a model when applied to unseen data.

For the **classification task**, we are presented with training data $S = (x_i, y_i)_{i=1}^m$ that consist of input features $x_i \in \mathcal{X}$ and corresponding labels $y_i \in \mathcal{Y}$. In this task, the goal is to obtain the lowest probability of classification error. Consequently, in this case, the objective function that needs to be minimized is the $0-1$ loss. This loss function counts how many errors a classifier makes. However, it is rarely used in optimization procedures because it is non-differentiable and non-continuous. To overcome this, many learning strategies use some convex *surrogates* of the $0-1$ loss function (*e.g.* hinge loss, squared error loss, cross-entropy).

Given a hypothesis set, a popular learning algorithm to find the best classifier in the hypothesis set is empirical risk minimization (ERM) introduced by Vapnik in Vapnik (1998), which minimizes the average loss on the given training data.

The objective of ERM is to find the model that, on average, produces the smallest loss on the training data. This is done by adjusting the model's parameters to reduce the training data's prediction error. Under the assumption of i.i.d data sampled from the distribution $\mathbb{P}_Z$, a substantial body of literature has been dedicated to derive generalization bounds that study the consistency of the classifier found by ERM (Bartlett et al., 1998b, 2006; Anthony and Bartlett, 1999; Muselli and Ruffino, 2004; Tewari and Bartlett, 2007; Biau et al., 2008a). Indeed, the consistency of classification algorithms plays a central role and is a well-explored topic in statistical learning theory. A consistent algorithm guarantees that taking more samples essentially suffices to reconstruct the unknown distribution roughly. This indicates that with enough training data, any desired level of performance can be attained. *However, in many real-world applications, due to flaws during the data collection and labeling process, the assumption that the training data is sampled from the true feature-label joint distribution does not hold.* Indeed, in real-world scenarios, classification datasets are often obtained through a labeling process usually done by humans. These labels might contain errors since they combine diverse individual annotations provided by numerous annotators who can disagree or make mistakes.

Motivation

As reported in Song et al. (2022) the ratio of corrupted labels in some real-world datasets is between 8.0% and 38.5%. Motivated by the presence of noisy labels in real scenarios for classification tasks, our investigation focuses on designing a more reliable model under such conditions.

The presence of inaccurate labels in the training dataset disrupts the assumption of consistency earlier mentioned, violating the underlying premises and undermining the performance guarantees provided by generalization bounds. This gap between theory and practice raises the question of whether it is possible to learn from datasets with noisy labels while still having performance guarantees.

Ideally, it would be desirable to design learning algorithms that are both robust to noisy labels and for which performance guarantees can be provided.

To answer this question, in Chapter 2 we design a method to build more robust models when multiple labels by different annotators are provided.

Providing theoretical guarantees for machine learning models addressing the three types of errors presented earlier is a key part of building more reliable models. Obtaining bounds for these errors, we increase the trustworthiness and effectiveness of our models in practical applications.

Being able to say which set of functions a model can approximate provides insights into which kind of underlying patterns in the data the model can capture. Knowing the model's approximation abilities aids in understanding where the model might excel and struggle, thereby preventing unrealistic assumptions and unexpected failures. Moreover, it helps in selecting the right model for a specific task. It ensures that the model is appropriate for the problem at hand, increasing its reliability in solving that particular task.

Improving optimization reduces the gap between the selected and the best possible model in the hypothesis class, making the models more reliable and accurate.

Reducing generalization error ensures that our models consistently provide accurate predictions in real-world applications. Indeed, generalization bounds offer insights into the behavior of a learning algorithm as the sample size increases, helping to understand the rate at which the error is expected to decrease with more data. *Understanding these bounds is essential for knowing how well a model can be expected to perform in practical applications and any possible conditions.*

In this research, we have focused on taking some steps in this direction in some specific cases.

In particular, we study the generalization error in the context of clustering, Chapter 3. These bounds describe the worst-case scenario, indicating that the error will be within a certain range given a specific number of samples. Sometimes, in practical experiments, better convergence results are encountered with respect to the theoretical results. This can be attributed to several factors, such as the characteristics of the data distribution, which may, in certain cases, exhibit more favourable properties than the worst-case distribution. Nevertheless, it is important to emphasize that these theoretical bounds remain valuable and insightful. Moreover, there are instances where the decrease rate of the generalization error is similar to that depicted by the theoretical bounds (see also the experiments reported in Chapter 3).

Furthermore, we *introduced a novel artificial neuron type designed to construct more interpretable networks.* Having an interpretable model enhances the user trustworthiness. *We also show that this type of network retains the same expressive power as standard neural networks* (i.e., it can approximate the same family of functions) and obtains performance on par with standard neural networks.

Finally, by characterizing the complexity of loss functions through topological analysis, we gain insights into the structure of loss landscapes. This understanding can be valuable in assessing the reliability of machine learning models since *the empirical loss landscape influences the gradient-descent optimization algorithms* that are typically used to find the best model fitting the data. For instance, a loss function exhibiting numerous local minima or a complex landscape could pose convergence and stability challenges. These issues can significantly impact the reliability of a model, as the struggle to navigate such a landscape may yield unpredictable or undesirable results. Therefore, the study of loss function topology becomes a valuable tool in ensuring the robustness and overall quality of machine learning models.

1.1 Research outline and questions

This thesis focuses on building reliable and trustworthy machine learning models by investigating generalization properties, the approximation capabilities of a specified hypothesis set, the robustness to the presence of noisy labels, and the topological complexity of loss functions. All these directions have generated multiple fruitful lines of work in machine learning and deep learning and continue to pose challenges to practitioners and theoreticians. The thesis is organized into four parts that reflect the flavour of the examples we presented earlier and the research directions followed during my PhD.

1.1.1 Leveraging inter-rater agreement for classification in the presence of noisy labels

Deep learning models have achieved state-of-the-art performance on various tasks, including image classification and generation (Khan et al., 2022; Ramesh et al., 2022; Krizhevsky et al., 2012), natural language processing (Chowdhary and Chowdhary, 2020; Khurana et al., 2023; OpenAI, 2023), and speech recognition (Prabhavalkar et al., 2023). However, their performance depends on the quality of the training data, prior studies have demonstrated deep learning models' susceptibility to overfit noisy labels when training data contains them, resulting in a degraded model performance (Zhang et al., 2016). In a practical setting, datasets can present noise; this can occur for various reasons: human error, the ambiguity of the task, or simply due to the existence of multiple correct labels for a single instance. A large body of work has been devoted to the challenge of learning with noisy labels, (Han et al., 2020; Song et al., 2022).

Often, in practical settings, classification datasets are obtained through a labelling process usually done by humans. In this case, labels can be noisy as they are obtained by aggregating the individual labels assigned to the same sample by multiple and possibly disagreeing annotators. This variability in annotation arises from differences in interpretation, subjectivity, or human error.

The inter-rater agreement can be measured to assess the quality and reliability of these labels by quantifying the extent to which annotators concur in their label assignments. While we can quantify the level of disagreement among annotators, the underlying noise distribution, which characterizes how these noisy labels are distributed across different categories or classes, remains unknown.

In Chapter 2 we aim to answer the following research questions (RQ):

> **RQ1** *How can inter-annotator statistics be leveraged in learning with noisy labels? Is there a better way to aggregate labels than majority vote?*

RQ2*Is it possible to learn from datasets with noisy labels while still having performance guarantees?*

Our contribution

We contributed to answering these questions in the following way

Estimation of Noise Distribution. We provide a methodology for estimating the noise distribution to which labels in classification datasets are subject. By leveraging inter-annotator statistics, it provides a novel way to approximate the noise transition matrix in the context of instance-independent noise, meaning that the corruption probability is so that $p(\tilde{y}|x, y) = p(\tilde{y}|y)$. The noise transition matrix is the matrix that describes the probability of true labels to be flipped, i.e. $T_{ij} = \mathbb{P}(\tilde{y} = j|y = i)$.

Methods for Learning from Noisy Datasets. We introduce a new method that utilizes the estimated noise distribution to improve learning from noisy datasets. The method takes advantage of the estimated noise distribution to improve learning. More in particular, for each sample and each class, an estimation of the posterior distribution of the sample's class given the annotations provided by different annotators is computed. The method uses the estimated noise transition matrix to compute the posterior distribution.

Generalization Bounds. We established generalization bounds within the empirical risk minimization framework for a well-known noise-robust loss function introduced in Patrini et al. (2017). This bound relies solely on quantifiable factors that can be practically estimated, in contrast to the generalization bounds of existing noise-tolerant training methods, which frequently hinge on unobservable variables, such as the true noise distribution.

Experimental Validation. We concluded by providing experimental results that illustrate the significance of the findings in practice. These experiments likely demonstrate the efficacy of the proposed methods for estimating the noise transition matrix and exploiting it in the learning process.

We go into more detail on this topic in Chapter 2. The work reported in that Chapter is partly based on Bucarelli et al. (2023a).

1.1.2 On generalization bounds for clustering

Given a set of points, clustering consists in finding a partition of a point set into clusters such that the center to which a point is assigned is as close as possible. Most commonly, centers are points themselves, which leads to the famous k-median and k-means objectives. One may also choose centers to be j dimensional subspaces, which gives rise to subspace clustering. We consider learning bounds for these problems. Generalization measures the difference between the empirical cost computed on a set of points P (training data) and the expected cost computed on the distribution $\mathcal{D}$ points are sampled from. It quantifies how well a model can adapt to new, unseen data. More in detail, Let P be a set of n points in d dimensional unit ball, and let k and z be positive integers. The objective of (k, z)- clustering consists of finding a set of k centers S minimizing $\text{cost}(P, S) := \sum_{p \in P} \min_{c \in S} \|p - c\|^z$. Given an unknown distribution $\mathcal{D}$ supported over the d-dimensional Euclidean unit sphere on the other hand, we define

$$\mathrm{cost}(\mathcal{D}, C) = \int_{B_2^d} \min_{c \in C} \|p - c\|^z \mathbb{P}[p] dp.$$

We denote by

$$C_{\mathcal{D}} = \underset{C}{\mathrm{argmin}}\ \mathrm{cost}(\mathcal{D}, C) \text{ and , } C_P = \underset{C}{\mathrm{argmin}}\ \mathrm{cost}(P, C).$$

RQ3 *How does $\mathbb{E}_P[cost(P, C_P)] - cost(\mathcal{D}, C_{\mathcal{D}})$ behave as function of k, d, and n for center-based clustering ?*

For the subspace clustering, we define the following quantities. Let P be a set of n points in d dimensional unit ball, and let k and j be a positive integer. The objective of (k, j, z)-clustering consists of finding a set of k subspaces $\mathcal{U}$ of dimension at most j minimizing

$$\mathrm{cost}(P, U) := \sum_{p \in P} \min_{V \in \mathcal{U}} \|(I - VV^T)p\|^z.$$

Given an unknown distribution $\mathcal{D}$ supported over the d-dimensional Euclidean unit ball

$$C_{\mathcal{D}} = \underset{C}{\mathrm{argmin}} \int_{B_2^d} \min_{U \in \mathcal{U}} \|(I - UU^T)p\|^z \cdot \mathbb{P}[p] dp \text{ and, } C_P = \underset{C}{\mathrm{argmin}}\ \mathrm{cost}(P, C).$$

RQ4 *How does $\mathbb{E}_P[cost(P, C_P)] - cost(\mathcal{D}, C_{\mathcal{D}})$ behave as function of k, d, j, and n for subspace clustering ?*

Our contribution

We helping to answer these questions, giving several near-optimal results. In particular,

For center-based objectives , we show a convergence rate of $\tilde{O}\left(\sqrt{k/n}\right)$. The $\tilde{O}$ notation hides logarithmic terms in the function providing the upper bound. This matches the known optimal bounds of Fefferman et al. (2016) and Bartlett et al. (1998a) for k-means and extends it to other important objectives such as k-median.

For subspace clustering with j-dimensional subspaces, we show that a convergence rate of $\tilde{O}\left(\sqrt{kj^2/n}\right)$. These are the first provable bounds for most of these problems. For the specific case of projective clustering, which generalizes k-means, we show a convergence rate of $\Omega\left(\sqrt{kj/n}\right)$ is necessary, thereby proving that the bounds from Fefferman et al. (2016) are essentially optimal.

We delve deeper into this subject in Chapter 3. The research presented in the chapter is partially derived from Bucarelli et al. (2023b).

1.1.3 A new generalization of the artificial neuron to enhance the interpretability of neural networks while preserving expressive capabilities

Neural Networks (NNs) have become the standard in most Artificial Intelligence (AI) applications, particularly deep neural networks. Studies in this domain have concentrated on enhancing the performance of NNs. However, as highlighted by Arrieta et al. (2020), achieving higher performance levels often involves a trade-off between model interpretability and accuracy. Recent efforts within the AI research community have been dedicated to the emergence of explainable artificial intelligence (XAI), a subfield focused on elucidating predictions derived from complex ML models. XAI has rapidly advanced, offering various methodologies to interpret AI model outputs, aiming to enhance their reliability and safety (Ghorbani et al., 2019) (Bhatt et al., 2019)). In this context, we propose NEWRON (Siciliano et al., 2022), a generalization of the McCulloch-Pitt neuron, allowing the definition of new artificial neurons. We show how special cases of NEWRON may pave the way towards interpretable, white-box neural networks. A fundamental property of neural networks is that of universal approximation. Under certain conditions, multilayer feed-forward neural networks can approximate any function in a given function space. Introducing this new kind of artificial neuron, the question arises whether networks developed from the novel neuron structure preserve this property. To explore this topic, we investigated the universality properties of diverse network types created with NEWRON. In Chapter 4, we're trying to answer the following question:

> **RQ5** *Is it possible to design a neural network structure that makes the whole model interpretable without sacrificing effectiveness and expressiveness?*

Our contribution

Interpret NEWRON-based networks. We present a method to interpret the behavior of NEWRON-based networks. In particular, we can devise exact rules when using the step function as activation. Also, in cases where from the network we cannot devise exact rules (e.g., in the sigmoid and tanh-prod cases), the structure of the neuron and the parameters allow the visualization of its behavior.

Universality. Through universal approximation theorems, we have proved that the new structure retains the same expressive power as a standard neural network.

Performances. Experiments on real and synthetic datasets illustrate how our framework can outperform traditional interpretable models, Decision Trees, and Logistic Regression and achieve similar or superior performance to standard neural networks.

The topic is further explored in Chapter 4. The content in that chapter draws in part from Siciliano et al. (2022).

1.1.4 A topological description of loss surfaces via Betti numbers characterization

Recent emphasis in deep learning research has centred on elucidating a suitable description of the loss function surface of deep neural network models. This focus aims to enhance comprehension of training methodologies using gradient descent-based methods. Despite the inherent non-convexity of associated optimization problems, deep neural networks are successfully trained using these methods. Prior works have made significant efforts to characterize the spurious minima for specific network architectures or to delineate the behavior of gradient dynamics. Our work contributes by seeking a topological characterization of the surface of the empirical risk of widely used loss functions in deep learning.

RQ6 *Can a topological measure effectively assess the complexity of the loss implemented by layered neural networks?*

RQ7 *How do the complexity bounds of deep and shallow neural architectures relate to the number of hidden units and the selected activation function?*

Our contribution

The study paves the way toward a more comprehensive understanding of the landscape of loss functions in deep learning models. This is done using the sum of Betti numbers of the sublevel sets of the loss function to measure the complexity of the empirical risk landscape. More precisely, we derived upper bounds on the complexity of empirical risk for deep and shallow neural architectures employing the theory of Pfaffian functions. This study illuminates the dependence of complexity on crucial factors such as the number of hidden units, the number of training samples, and the specific choice of activation function.

We offer a more comprehensive discussion of this work in Chapter 5. The content covered in the chapter primarily stems from unpublished research conducted in collaboration with Giuseppe Alessio D'Inverno, Monica Bianchini and Franco Scarselli from the University of Siena.

1.2 Thesis overview

This thesis is organized into four parts; each can be read independently. The structure is designed in alignment with the problems and topics mentioned earlier. The first part, Chapter 2, focuses on mitigating the impact of noisy labels. In this context, we introduce an innovative strategy that harnesses inter-rater agreement and noise distribution estimation, aiming to ameliorate model performance and robustness. The results of this chapter lean on Bucarelli et al. (2023a). In the second part, Chapter 3), we establish nearly optimal generalization bounds for clustering and projective clustering; this chapter is based on Bucarelli et al. (2023b). In the third part of the thesis, Chapter 4, we introduce a novel artificial neuron that amplifies interpretability while preserving the expressive power of standard neural network models. We observed experimentally that models based on this artificial neuron obtain better performance compared to popular interpretable by design models and are on par with standard neural networks. This chapter is based on Siciliano et al. (2022). Finally, the fourth part, Chapter 5, is devoted to studying a characterization of the Betti numbers of the loss landscape. We derived upper bounds on the complexity in terms of the sum of the Betti number of level sets defined by the loss function of neural architectures. The last work is mostly based on a project not yet published, a collaboration with Giuseppe Alessio D'Inverno, Monica Bianchini and Franco Scarselli from the University of Siena.

Chapter 2

Leveraging Inter-rater Agreement for Classification in the Presence of Noisy Labels

In recent years, artificial intelligence (AI) has witnessed remarkable advancements across various tasks, including natural language understanding and computer vision (He et al., 2016; Devlin et al., 2018; OpenAI, 2023; Khan et al., 2022; Ramesh et al., 2022). Yet, these models heavily depend on the quality of their training data. When training labels are corrupted, deep learning models can excessively adjust to these inaccuracies, impacting their performance. However, the effectiveness of these models is heavily reliant on the quality of the data used for training. When the training labels are corrupted, deep learning models become susceptible to adapting too closely to the noisy labels. This, in turn, compromises the models's overall performance.

The introduction of label noise in training data can be attributed to various factors, including human errors during data labeling, inherent ambiguities in the learning task, or multiple valid labels for a single data instance.

Multiple studies have presented learning algorithms capable of managing datasets containing corrupted labels. These algorithms ensure satisfactory performance by leveraging established generalization boundaries. However, these remedies do not comprehensively resolve the issue, as they hinge on accurate knowledge of the error rate affecting the labels. Unfortunately, this rate is frequently elusive in real-world applications. Several endeavours (Patrini et al., 2017; Xia et al., 2019; Yao et al., 2020) strive to tackle this challenge by introducing methodologies for estimating this error rate. Nonetheless, some of these techniques suffer from the drawback of relying on assumptions that are not universally applicable, such as the assumption of anchor samples (Patrini et al., 2017). Ideally, it would be desirable to design learning algorithms that are both robust to noisy labels, and for which performance guarantees can be provided.

An alternative strategy, commonly employed in industry to mitigate the impact of errors introduced by human annotators, involves labeling the same dataset mutliple times using different raters. Subsequently, the individual labels are merged to diminish the likelihood of erroneous labels within the dataset. Two prevalent techniques for this purpose are majority vote and soft labeling. In such scenarios, metrics like inter-annotator agreement (IAA) scores (examples include Cohen's kappa (Cohen, 1960) and Fleiss' kappa (Fleiss et al., 1971)) furnish quantifiable measurements that directly correspond to the presence of error probabilities within the labels. This approach differs primarily in that it addresses the labeling process instead of focusing on alterations to the learning algorithm. Because the inter-annotator agreement (IAA) is intrinsically tied to the error rate linked with human annotators, it becomes plausible to approximate this error rate. Subsequently, this estimation can be employed to adapt the learning algorithms, aiming to enhance their resilience against the resultant label noise. This constitutes the primary avenue of investigation in our study. In other words, in this Chapter, we aim to answer the following questions:

> **RQ1** *How can inter-annotator statistics be leveraged in learning with noisy labels? Is there a better way to aggregate labels than majority vote?*
>
> **RQ2***Is it possible to learn from datasets with noisy labels while still having performance guarantees?*

The new contributions provided in this work are the following:

i) we provide a methodology to estimate the label noise distribution based on the IAA statistics;

ii) we show how to leverage this estimate to learn from the noisy dataset;

iii) we provide generalization bounds for our methods that depend on **known** quantities.

2.1 Related works

Our work is related to literature on three main topics: (i) robust loss function design, (ii) label aggregating and (iii) noise rate estimation.

Robust loss functions Ghosh et al. (2017) and Ghosh et al. (2015) shown that *symmetric* loss functions, that are functions for which the sum of the risks over all categories is equivalent to a constant for each arbitrary example, are robust to label noise. Examples of symmetric loss functions include the $0 - 1$ loss, the Ramp Loss and (softmax) Mean Absolute Error (MAE). In Zhang and Sabuncu (2018) authors show that even if MAE is noise tolerant and cathegorical cross entropy (CCE) is not, MAE can perform poorly when used to train DNN in challenging domains. They also propose a loss function that can be seen as a generalization of MAE and CCE. Several other loss functions that do not strictly satisfy the symmetry condition have also been proposed to be robust against label noise when training deep neural networks (Menon et al., 2020; Wang et al., 2019b; Feng et al., 2020).

Natarajan et al. (2013) presents two methods to modify the surrogate loss in the presence of class-conditional random label noise. The first method introduces a new loss that is an unbiased estimator for a given surrogate loss, and the second method introduces a label-dependent loss. The paper provides generalization bounds for both methods, which depend on the noise rate of the dataset and the complexity of the hypothesis space.

Labels aggregation When constructing datasets for supervised learning, data is often not labelled by a single annotator, rather multiple imperfect annotators are asked to assign labels to documents. Typically, separate labels are aggregated into one before learning models are applied (Raykar et al., 2010; Dawid and Skene, 1979). In our work, we propose to exploit a measure of the agreement between annotators to explicitly calculate the noise of the dataset. Recently some works revisited the choice of aggregating labels. In Purpura et al. (2022) authors explore how to train LETOR models with relevance judgments distributions instead of single-valued relevance labels. They interpret the output of a LETOR model as a probability value or distribution and define different KL divergence-based loss functions to train a model. The loss they proposed can be used to train any ranking model that relies on gradient-based learning (in particular they focused on transformer-based neural LETOR models and on the decision tree-based GBM model). However, the authors do not directly estimate the noise rates in the annotations or study how learning from these noisy labels affects the generalization error of the models trained with the methods they introduce. In Wei et al. (2022) the authors analyze the performance of both label aggregation and non-aggregation approaches in the context of empirical risk minimization for a number of popular loss functions, including those designed specifically for the noisy label learning problem. They conclude that label separation is preferable to label aggregation when noise rates are high or the number of labellers/annotations is insufficient. Peterson et al. (2019) and Uma et al. (2020) exploit the availability of multiple human annotations to construct soft labels and concludes that this increases performance in terms of generalization to out-of-training-distribution test datasets and robustness to adversarial attacks. Collins et al. (2022) focus on efficiently eliciting soft labels from individual annotators.

Noise rate estimation A number of approaches have been proposed for estimating the noise transition matrix (i.e. the probabilities that correct labels are changed for incorrect ones) (Patrini et al., 2017; Menon et al., 2015; Zhu et al., 2022). Usually, these methods use a small number of anchor points (that are samples that belong to a specific class with probability one) (Hendrycks et al., 2018). In particular, (Patrini et al., 2017) proposed a noise estimation method based on anchor points, with the intent to provide an 'end-to-end' noise-estimation-and-learning method. Due to the lack of anchor points in real data, some works focused on a way to detect anchor points in noisy data, (Yao et al., 2020; Xia et al., 2019). In Yao et al. (2020) the authors propose to introduce an intermediate class to avoid directly estimating the noisy class posterior. Zhang et al. (2021) also propose an iterative noise estimation heuristic that aims to partly correct the error and pointed out that the methods introduced by Patrini et al. (2017) and Yao et al. (2020) have an error in computing anchor points, and provide conditions on the noise under which the methods work or fail. (Xia et al., 2019) provides a solution that can infer the transition matrix without anchor points. Indeed they use the instances with the highest class posterior probabilities for noisy data as anchor points. Our work differs from the mentioned work that uses anchor points because we do not need to assume the existence of anchor points or to have a validation set to learn the noise rate and we only use noisy data to train our model, moreover we neither aim to detect anchor points in the noisy data. Also, most of these works do not study the generalization properties of the proposed models, while we also address this problem and find bound that depend on the estimated noise transition matrix.

Another approach is based on the clusterability condition, that is an example belongs to the same true class of its nearest-neighbors representations. Zhu et al. (2021) presented a method that relies on statistics of high-order consensuses among the 2 nearest-neighbors noisy labels.

2.2 Problem formulation

2.2.1 Notation

In this chapter, we follow the following notation. Matrices and sets are denoted by upper-case and calligraphic letters, respectively. The space of d-dimensional feature vectors is denoted by $\mathcal{X} \subset \mathbb{R}^d$.

We denote by C the number of classes and by e_j the j-th standard canonical vector in $\mathbb{R}^C$, namely the vector that has 1 in the j-th position and zero in all the other positions. $\mathcal{Y} = \{e_1, \dots, e_C\} \subset \{0,1\}^C$ is the label set. Feature vectors and labels are denoted by x and y, respectively. $\mathcal{D}$ is the joint distribution of the feature vectors and labels, i.e. $(x, y) \sim \mathcal{D}$. The sampled dataset of size n is denoted by $\widehat{\mathcal{D}} = \{(x_i, y_i)\}_{i=1}^n$. $f(x)$ denotes the output of the classifier f for feature vector x and is a C dimensional vector. All vectors are column vectors.

We denote by $\ell(t, y)$ a generic loss function for the classification task that takes as input C dimensional vectors t and y. In practice t will contain the prediction of the model, and y will be the ground-truth label as a one-hot encoded vector. Namely $\ell : [0,1]^C \times \mathcal{Y} \to \mathbb{R}$.

2.2.2 Background

We consider the classification problem within the supervised learning framework, where the ultimate goal is to minimize the ℓ -risk $R_{\ell,\mathcal{D}}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f(x), y)]$, for some loss function ℓ. We denote by $\mathcal{D}$ the joint distribution of feature vectors x and labels y. In practice, since the distribution is unknown instead of minimizing $R_{\ell,\mathcal{D}}(f)$ we minimize an empirical risk over some sampled dataset $\widehat{\mathcal{D}}$:

$$\widehat{R}_{\ell,\widehat{\mathcal{D}}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i) = \mathbb{E}_{(x,y)\sim\widehat{\mathcal{D}}}[\ell(f(x), y)]. \tag{2.1}$$

In this work, we assume that the true labels y_i are unknown and consider two scenarios, both of which rely on H annotators.

Scenario I

In this scenario we have access to the H labels provided by the annotators for each sample, where $y_{i,a}$ refers to the label provided by the a-th annotator for the i-th sample. For a given feature vector x_i the distribution of labels provided by annotator a is given by its noise transition matrix T_a, which is defined as follows:

$$(T_a)_{i,j} := \mathbb{P}(y_a = j | y = i) \tag{2.2}$$

Assumption 1. *We assume that all annotators have the same noise transition matrix (i.e. $T_a = T$ for all a), that T is symmetric and that its diagonal elements are larger than* 0.5 *(i.e. $\mathbb{P}(y_a = i | y = i) > 0.5, \forall i \in \{1, \dots C\}$).*

Note that by definition T is right stochastic and hence also doubly stochastic. It is also strictly diagonally dominant and therefore non-singular.

Proposition 2.2.0.1. *T is positive definite.*

Proof. Since T is symmetric it follows that all eigenvalues are real. Combining the fact that it is strictly diagonally dominant with Gershgorin's theorem we conclude that all eigenvalues lie in the range $(0, 1]$ and hence T is positive definite. □

Assumption 2. *We assume that the annotators are conditionally independent on the true label y:*

$$\mathbb{P}(y_a, y_b|y) = \mathbb{P}(y_a|y)\mathbb{P}(y_b|y). \tag{2.3}$$

We now define the IAA matrix M_{ab} between annotators a and b as follows:

$$(M_{ab})_{i,j} := \mathbb{P}(y_a = i, y_b = j) \tag{2.4}$$

Proposition 2.2.0.2. *Leveraging Assumption 2 the agreement matrix $M_{a,b}$ can be written as follows:*

$$M_{a,b} = {T_a}^T D T_b \tag{2.5}$$
$$D := diag\{\nu\} \tag{2.6}$$
$$\nu := [\mathbb{P}(y = 1), \cdots, \mathbb{P}(y = C)]^T. \tag{2.7}$$

Due to Proposition 2.2.0.1 and the fact that D is positive definite, it follows that all matrices $M_{a,b}$ are invertible.

Assumption 3. *We assume that the class probabilities (and hence D) are known.*

Due to Assumption 1, all annotators share the same noise transition matrix T. Therefore M_{ab} is independent of a and b, and from now on, we remove this dependency in the notation(i.e. we get $M = T^T DT$). Furthermore, since T is invertible and D is diagonal and positive definite, it follows that M is also positive definite.

Note that since we have access to all the labels provided by the H annotators for all the samples, we can obtain an estimate of M which we denote $\widehat{M}$.

Assumption 4. *We assume that $\widehat{M}$ is a consistent estimator.*

For the case of two annotators, one possible consistent estimator $\widehat{M_{a,b}}$ that exploits its symmetry condition is given by:

$$(\widehat{M_{a,b}})_{i,j} = \sum_{k=1}^{n} \frac{\mathbb{1}(y_{a,k}{=}i, y_{b,k}{=}j) + \mathbb{1}(y_{a,k}{=}j, y_{b,k}{=}i)}{2n} \tag{2.8}$$

If the annotators have the same transition matrix, M will be the same for all pairs of annotators. So we can estimate M, in the case of $H \geq 2$ by averaging the estimators $\widehat{M_{ab}}$ obtain by Eq. (2.8) for all possible pairs of annotators. The estimator in this case can be written as

$$(\widehat{M})_{i,j} = \frac{1}{H(H{-}1)} \sum_{a=1}^{H} \sum_{\substack{b=1 \\ b\neq a}}^{H} \sum_{h=1}^{n} \frac{\mathbb{1}(y_{a,h}{=}i, y_{b,h}{=}j)}{n}. \tag{2.9}$$

Scenario II

In the second scenario, for each i-th sample we are given a unique label $\tilde{y}_i$ that is produced by aggregating the H individual labels according to some known aggregating policy (like majority vote). In this case, since we do not have access to the individual annotations we assume that $\widehat{M}$ is provided.

The probability that label y_i is corrupted to some other label $\tilde{y}_i$ is given by the *aggregated noise transition matrix* $\Gamma \in [0,1]^{C\times C}$, where $\Gamma_{ij} := \mathbb{P}(\tilde{y} = j|y = i)$ is the probability of the true label i being flipped into a corrupted label j and C is the number of classes. Note that by definition Γ is a right stochastic matrix that is determined by T, the amount of annotators H and the aggregating policy. We will study both the case where $\Gamma = T$, and the case in which there exists a generic Lipschitz function ϕ so that $\Gamma^{-1} = \phi(T)$.

There are different policy choices to construct the dataset that lead to $\Gamma = T$. If we decide to use only one annotator, for instance a, to build the final dataset, namely for each sample $\tilde{y}^i = y^i_a$ we have $\Gamma = T_a$. Or if annotators are homogeneous, i.e. they have the same noise transition matrix T, and to build the final dataset we decide to randomly select the label of one of the annotators we have that $\Gamma = T$.

Even restricting ourselves to the case of homogeneous annotators, depending on the rule with which we build the dataset we can have a more complex relationship between the matrix T and Γ.

We also obtain generalization bounds in the case were an estimate of the agreement matrix M is not available and we only have access to a scalar representation of the inter-annotator agreement, in particular we consider the case where the Cohen's κ is given.

Objective

The objective in both scenarios is to: i) use $\widehat{M}$ to estimate the noise transition matrices (T and Γ); ii) leverage these estimates to be able to learn from the noisy dataset in a more robust manner; and iii) obtain generalization bounds for the resulting learning methods.

2.3 Main results

We divide the main contributions in three sections. In the first section we show how to estimate the noise matrices T Next we indicate how to leverage these estimates to learn for the datasets with noisy labels. Finally we obtain bounds,depending on the Rademacher complexity of the class of functions, on the generalization gap for a bounded and Lipschitz loss function

2.3.1 Estimation of the noise transition matrices

We start stating the following Lemma that allows us to write the unknown matrix T (and its inverse), as a function of D and M.

Lemma 2.3.1. *If $D^{\frac{1}{2}}$ commutes with T we have that:*

$$T = U\Lambda^{\frac{1}{2}}U^T \tag{2.10}$$

$$T^{-1} = U\Lambda^{-\frac{1}{2}}U^T \tag{2.11}$$

$$D^{-\frac{1}{2}}MD^{-\frac{1}{2}} = U\Lambda U^T \tag{2.12}$$

where $U\Lambda U^T$ is the eigenvalue decomposition of $D^{-\frac{1}{2}}MD^{-\frac{1}{2}}$ (i.e. U is some orthogonal matrix and Λ is a diagonal positive definite matrix).

Proof. From Eq. (2.5) we get:

$$M = TDT = D^{\frac{1}{2}}TTD^{\frac{1}{2}} \rightarrow D^{-\frac{1}{2}}MD^{-\frac{1}{2}} = T^2 \tag{2.13}$$

Note that T and $D^{\frac{1}{2}}MD^{\frac{1}{2}}$ are positive definite (because D and M are positive definite) and hence they have eigenvalue decompositions of the following form:

$$T = U_T\Lambda_T U_T^T \tag{2.14}$$
$$D^{-\frac{1}{2}}MD^{-\frac{1}{2}} = U_M\Lambda_M U_M^T \tag{2.15}$$

where U_x are orthogonal matrices and Λ_x are diagonal positive definite matrices. It then follows that:

$$T^2 \overset{(a)}{=} U_T\Lambda_T^2 U_T^T = U_M\Lambda_M U_M^T \tag{2.16}$$

where in (a) we used the fact that U_T is orthogonal. Since $U_M\Lambda_M U_M^T$ is an eigenvalue decomposition of T^2 we conclude that:

$$T = U_M\Lambda_M^{\frac{1}{2}}U_M^T, \quad T^{-1} = U_M\Lambda_M^{-\frac{1}{2}}U_M^T \tag{2.17}$$

□

Note that we could use Lemma 2.3.1 to estimate T as follows:

$$\widehat{T} = \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T \tag{2.18}$$

where $\widehat{U}\widehat{\Lambda}_M\widehat{U}^T$ is the eigenvalue decomposition of $D^{-\frac{1}{2}}\widehat{M}D^{-\frac{1}{2}}$. However such estimate can result in matrices that are not doubly stochastic, or diagonally dominant due to estimation errors. A more accurate estimate of T could be obtained as $\widehat{T} = \pi(\widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T)$ where π is a projection operator to the set of doubly stochastic, positive definite matrices with diagonal elements greater than 0.5 and non-negative entries (which is a convex set). We can obtain such projection by solving the following optimization problem:

$$\widehat{T} = \pi(\widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T) = \underset{B}{\operatorname{argmin}}||B - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2 \tag{2.19}$$

$$\text{s.t.} \qquad \begin{aligned} &B = B^T \\ &\sum_j B_{i,j} = 1 \quad \forall i \\ &B_{i,j} \geq 0 \quad \forall i,j \\ &B_{i,i} \geq 0.5 \quad \forall i \end{aligned}$$

Note that this optimization problem is convex because the constraints are linear and for symmetric matrices it holds that $||\widehat{T} - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2 = \lambda_{\max}(\widehat{T} - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T)$, which is a convex function of $\widehat{T}$.

To summarize, T can be estimated as follows. First, obtain an estimate of M. Then obtain the eigenvalue decomposition of $D^{-\frac{1}{2}}\widehat{M}D^{-\frac{1}{2}} = \widehat{U}\widehat{\Lambda}\widehat{U}^T$ (note that this decomposition always exists because $D^{-\frac{1}{2}}\widehat{M}D^{-\frac{1}{2}}$ is symmetric). Finally obtain the estimate as: $\widehat{T} := \pi(\widehat{U}\widehat{\Lambda}^{\frac{1}{2}}\widehat{U}^T)$.

Note that once the estimate of $\widehat{T}$ is obtained, $\widehat{\Gamma}$ can be obtained since we assumed the label aggregating policy to be known.

We are now presenting a concise overview of when the commutativity assumption holds.

2.3.2 On the hypothesis of commutativity in Lemma 2.3.1

In the previous we found how to compute T given M and D. To find this relationship we require that $D^{\frac{1}{2}}$ commutes with T. This hypothesis is satisfied when D and T have a particular structure, namely

$$\frac{\sqrt{d_i}}{\sqrt{d_j}}t_{ij} = t_{ij} \quad \forall i \text{ and } j.$$

That is satisfied or if $d_i = d_j$ or if $t_{ij} = 0$, namely every class so that the probability of going from class i to class j (and vice-versa) is not zero is equiprobable.

So T has to be block diagonal, or better reducible by a permutation of the classes to a block diagonal matrix and D has to have all equal elements on indices relatives to the same block in T. For instance

$$T = \begin{pmatrix} T_1 & 0 & 0 & 0 & 0 \\ 0 & T_2 & 0 & 0 & 0 \\ 0 & 0 & T_3 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & T_j \end{pmatrix} \text{ and } D = \begin{pmatrix} D_1 & 0 & 0 & 0 & 0 \\ 0 & D_2 & 0 & 0 & 0 \\ 0 & 0 & D_3 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & D_j \end{pmatrix}$$

with

$$D_i = \begin{pmatrix} d_i & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & d_i \end{pmatrix}$$

T need not be block diagonal but must be reconducted to a block diagonal matrix by permuting the classes, for instance in the following case, we can obtain a matrix block diagonal by permuting classes 2 and 4

$$T = \begin{pmatrix} t_{11} & 0 & 0 & t_{14} \\ 0 & t_{22} & t_{23} & 0 \\ 0 & t_{23} & t_{33} & 0 \\ t_{14} & 0 & 0 & t_{44} \end{pmatrix} \quad \text{and} \quad D = \begin{pmatrix} d_1 & 0 & 0 & 0 \\ 0 & d_2 & 0 & 0 \\ 0 & 0 & d_2 & 0 \\ 0 & 0 & 0 & d_1 \end{pmatrix}$$

Notice that T can be rewritten as follows permuting classes 2 and 4

$$T = \begin{pmatrix} t_{11} & t_{14} & 0 & 0 \\ t_{14} & t_{44} & 0 & 0 \\ 0 & 0 & t_{33} & t_{23} \\ 0 & 0 & t_{23} & t_{22} \end{pmatrix}$$

From the technical point of view, we have noticed that solving this equation is extremely complicated without making such assumptions. Another assumption we could have used, also required by Potter (1966) to solve the same problem, is requiring that the matrix $D^{\frac{1}{2}}T$ has diagonal Jordan decomposition. However, this assumption is more complicated to translate at the level of the structure of the matrices T and D.

From a practical point of view, making such an assumption means that there are classes that annotators can confuse with one another while they never swap between them, other classes. For example, if the problem is to classify images and the classes are “cat”, “lynx”, “bats”, “bird”, “cougar”; we can think that the annotators have a non-zero probability of confusing with each other the feline classes “lynx”, “cat”, “cougar”, while they have zero probability of assigning a picture of a lynx the label “bird”. Commutativity is guaranteed in the case of a uniform distribution over the classes. There are many applications where we expect the distribution over the classes to be uniform and not to have any class with a higher probability. In general, we can fall back to an approximation of this case by reducing the samples.

Lemma 2.3.2. *Let $M_{a,b}$ be the agreement matrix for annotators a and b defined in Eq. (2.4) and $\widehat{M_{a,b}}$ be the estimated agreement matrix defined in Eq. (2.8) and let $||.||_p$ be the matrix norm induced by the p vector norm. For every $p \in [1, \infty]$ and for every $\delta > 0$, with probability at least $1 - \delta$*

$$||M_{a,b} - \widehat{M_{a,b}}||_p \leq \sqrt{\frac{C^2}{2n} \ln \frac{2C^2}{\delta}}. \tag{2.20}$$

where $\mathbb{P}^n$ denotes the probability according to which the n training samples are distributed, i.e. we are assuming that the samples are independently drawn according the probability $\mathbb{P}$.

To prove Lemma 2.3.2 we need the following Propositions and Lemma.

Proposition 2.3.2.1. *Let $M_{a,b}$ be the agreement matrix for annotators a and b defined in Eq. (2.4) and $\widehat{M_{a,b}}$ be the estimated agreement matrix defined in eq. Eq. (2.8). For every $\epsilon > 0$ it holds that*

$$\mathbb{P}^n(|(M_{a,b})_{ij}) - (\widehat{M_{a,b}})_{ij}| < \epsilon) \geq 1 - 2e^{-2\epsilon^2 n}.$$

And

$$\mathbb{P}^n\Big(\forall i, j \in \{1, C\}^2 \; |(M_{a,b})_{ij}) - (\widehat{M_{a,b}})_{ij}| < \epsilon\Big) \geq 1 - 2C^2 e^{-2\epsilon^2 n}.$$

where $\mathbb{P}^n$ denotes the probability according to which the n training samples are distributed, i.e. we are assuming that the samples are independently drawn according to the probability $\mathbb{P}$.

To simplify the notation we will omit the dependency from the annotators in the matrices: $M = M_{a,b}$ and $\widehat{M} = \widehat{M}_{a,b}$ $M_{ij} = \mathbb{P}(y_a = i, y_b = j)$ and $\widehat{M}_{ij} = \frac{1}{n}\sum_{h=1}^{n} \mathbb{1}((y_a)_h = i, (y_b)_h = j)$.

Proof. To prove the claim we only need to apply Hoeffding’s inequality to the random variables $X_h^{ij} = \mathbb{1}_{{y_a}_h = i, = {y_b}_h = j}$. Indeed it holds that $0 \leq X^{ij} \leq 1$ and $\widehat{M}_{ij} = \frac{1}{n}\sum_{h=1}^{n} X_h^{ij}$, while $\mathbb{E}[X_h^{ij}] = M_{ij}$.

Notice that the random variables $X_1^{ij} \dots X_n^{ij}$ are independent since we assume samples to be independent with respect to each other and so it follows that $(x_h, y_{a_h}, y_{b,h}), (x_k, y_{a_k}, y_{b_k})$ are independent.rf

$$\mathbb{P}(|\mathbb{E}[X_h^{ij}] - \frac{1}{n}\sum_{h=1}^{n} X_h^{ij}| > \epsilon) \leq 2e^{-2\epsilon^2 n}. \tag{2.21}$$

From the previous equation, using union bounds we can obtain that

$$\mathbb{P}\Big(\forall (i,j) \in \{1, C\}^2 \; |\mathbb{E}[X_h^{ij}] - \frac{1}{n}\sum_{h=1}^{n} X_h^{ij}| < \epsilon \Big) \geq 1 - 2C^2 e^{-2\epsilon^2 n}. \tag{2.22}$$

Namely

$$\mathbb{P}\Big(\forall (i,j) \in \{1, C\}^2 \; |M_{ij} - \widehat{M}_{ij}| < \epsilon \Big) \geq 1 - 2C^2 e^{-2\epsilon^2 n}. \tag{2.23}$$

□

Lemma 2.3.3. *Let A be a matrix in $\mathbb{R}^{CxC}$ so that it exists $\epsilon > 0$ for all i, j $|A_{ij}| \leq \epsilon$. For every $p \in [1, \infty]$, if $||.||_p$ denotes the matrix norm induced by the p-vector norm,*

$$||A||_p \leq C\epsilon.$$

Proof.

$$||A||_p := \sup_{x : ||x||_p = 1} ||Ax||_p$$

Let x be a vector of p-norm 1. $(Ax)_i = \sum_{j=1}^{C} A_{ij} x_j$

$$||Ax||_p = \Big(\sum_{i=1}^{C}\Big|\sum_{j=1}^{C} A_{ij}x_j\Big|^p\Big)^{\frac{1}{p}} \leq \Big(\sum_{i=1}^{C}\Big(\sum_{j=1}^{C}|A_{ij}x_j|\Big)^p\Big)^{\frac{1}{p}} \leq \epsilon\Big(\sum_{i=1}^{C}\Big(\sum_{j=1}^{C}|x_j|\Big)^p\Big)^{\frac{1}{p}}$$

Now, denoting by $\mathbf{1}$ the vector with all ones, using Hölder inequality we can obtain :

$$\sum_{j=1}^{C} |x_j| = ||\mathbf{1}x||_1 \leq ||x||_p ||\mathbf{1}||_{\frac{p}{p-1}} = ||x||_p C^{\frac{p-1}{p}}$$

So

$$||Ax||_p \leq \epsilon\Big(\sum_{i=1}^{C} ||x||^p C^{p-1}\Big)^{\frac{1}{p}} = \epsilon C ||x||_p = \epsilon C$$

□

Proof Lemma 2.3.2. For the previous Lemma it holds that if all the elements of the matrix are less or equal than ϵ, the p norm is bounded by ϵC

So we can derive that

$$\mathbb{P}(||M_{a,b} - \widehat{M}_{a,b}||_p > \epsilon) \geq \mathbb{P}\Big(\forall (i,j) \in \{1, C\}^2 \; |M_{ij} - \widehat{M}_{ij}| < \frac{\epsilon}{C} \Big) \geq 1 - 2C^2 e^{-2\frac{\epsilon^2}{C^2} n}. \tag{2.24}$$

□

From Lemma 2.3.2 it follows that if $\widehat{M}$ is estimated as in Eq. (2.9), since $\widehat{M}$ is an average of $\widehat{M_{ab}}$ it also holds that for every $p \in [1, \infty]$ and for every $\delta > 0$, with probability at least $1 - \delta$

$$||M - \widehat{M}||_p \leq \sqrt{\frac{C^2}{2n} \ln \frac{2C^2}{\delta}}. \tag{2.25}$$

Theorem 2.3.4. *Let T be the noise transition matrix defined as in Eq. (2.2) and $\widehat{T}$ its estimate (defined as in Eq. (2.19)).*

With probability at least $1 - \delta$:

$$||T - \widehat{T}||_2 \leq \frac{C(\sqrt{C}+1)\lambda_{\max}(D)}{\lambda_{\min}(\widehat{T})} \sqrt{\frac{1}{2n} \ln \frac{2C^2}{\delta}} \tag{2.26a}$$

$$||T^{-1} - \widehat{T}^{-1}||_2 \leq \frac{9C(\sqrt{C}+1)\lambda_{\max}(D)}{\lambda_{\min}(\widehat{T})^2} \sqrt{\frac{1}{2n} \ln \frac{2C^2}{\delta}} \tag{2.26b}$$

for $n > \frac{C^2(\sqrt{C}+1)^2(\ln(2C^2)^2}{2\lambda_{\min}(\widehat{T})^2}$.

Proof of Theorem 2.3.4: bound error on the estimation of T

We start by introducing the following helpful remark and Lemmas.

Remark 1. *We defined $\widehat{T} = \underset{B}{\operatorname{argmin}} ||B - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2$, with B that satisfies all the constraints in Eq. (2.20). We know that the matrix T we want to approximate satisfies all the constraints in Eq. (2.20), so by definition*

$$||\widehat{T} - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2 \leq ||T - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2,$$

from which it follows that

$$||T - \widehat{T}||_2^2 \leq 2||T - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2$$

so any bound we will found for $||T - \widehat{U}\widehat{\Lambda}_M^{\frac{1}{2}}\widehat{U}^T||_2^2$ holds also for $\widehat{T}$ estimated as in Eq. (2.19) with a coefficent 2.

Lemma 2.3.5. *Let A be a square, symmetric, positive definite matrix, in $\mathbb{R}^{C \times C}$ and let $\sqrt{A}$ the unique positive definite symmetric, matrix so that $\sqrt{A}\sqrt{A} = A$ (On the existence of this matrix, see Theorem 7.2.6 at p. 439 in Horn and Johnson (2012)). The bounded operator $F_{\sqrt{}} : \mathcal{S} \to \mathcal{S}$ defined as follow $F_{\sqrt{}} : A = \sqrt{A}$, where we denote by $\mathcal{S}$ the space of symmetric positive definite matrix, is differentiable and it hold the following upper bound for the induced 2 norm of the derivative*

$$||D[\sqrt{A}]||_2 \leq \frac{1}{2\sqrt{\lambda_{\min}(A)}} ||vec(A)||_2. \tag{2.27}$$

Proof. Let us consider the vector space of square matrices $M_C(\mathbb{R})$ with the 2 norm and let $D[\sqrt{A}]$ denote the operator that is the derivative of $F_{\sqrt{}}$ in this space and $D[A]$ the derivative of A. From the fact that $\sqrt{A}\sqrt{A} = A$ it follows that

$$D[\sqrt{A}]\sqrt{A} + \sqrt{A}D[\sqrt{A}] = D[A]. \tag{2.28}$$

Eq. (2.28) is a special case of Sylvester equation, and using that $\sqrt{A}$ is symmetric can be rewritten as

$$(I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)\text{vec}(D[\sqrt{A}]) = \text{vec}(D[A]). \tag{2.29}$$

It follow that

$$\text{vec}(D[\sqrt{A}]) = (I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)^{-1}\text{vec}(D[A]) = (I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)^{-1}\text{vec}(A).$$

Notice that the eigenvalues of the square root of a symmetric, positive def matrix are the square root of the eigenvalues of the original matrices. Indeed if A can be decomposed as $A = U\Lambda U^T$, with U orthogonal matrix, it holds that $\sqrt{A} = U\sqrt{\Lambda}U^T$. Now the eigenvalues of $\sqrt{A} \otimes I_C + I_C \otimes \sqrt{A}$ are $\sqrt{\lambda_i} + \sqrt{\lambda_j}$ with $1 \leq i, j \leq C$, with λ_i eigenvalue of A. The minimum eigenvalue of a symmetric positive def matrix B is the maximum eigenvalue of the inverse, indeed if $B = VDV^T$, with V orthogonal, $B^{-1} = VD^{-1}V^T$. So the minimum eigenvalue of $\sqrt{A} \otimes I_C + I_C \otimes \sqrt{A}$, that is the maximum eigenvalue of $(\sqrt{A} \otimes I_C + I_C \otimes \sqrt{A})^{-1}$ is $2\lambda_{\min}(\sqrt{A})$. It follows that

$$\begin{aligned} ||(I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)^{-1}||_2 &= \sqrt{\lambda_{\max}((I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)^{-2})} \\ &= \sqrt{\lambda_{\min}((I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)^{2})} \\ &= \lambda_{\min}((I_C \otimes \sqrt{A} + \sqrt{A} \otimes I_C)) \\ &= \frac{1}{2\sqrt{\lambda_{\min}(A)}}. \end{aligned}$$

So $||\text{vec}(D[\sqrt{A}])||_2 \leq \frac{1}{2\sqrt{\lambda_{\min}(A)}}||\text{vec}(A)||_2$. $||\text{vec}(A)||_2^2 = \sum_{k=1}^{C^2} a_k^2$ for every vecto x of norm 1 (this implies $x_i < 1$)

$$||Ax||_2^2 = \sum_{k=1}^{C}\sum_{i=1}^{C} a_{ki}^2 x_i^2 \leq \sum_{k=1}^{C}\sum_{i=1}^{C} a_{ki}^2 = ||\text{vec}(A)||_2^2.$$

It follows that the induce 2 norm of the derivative $||D[\sqrt{A}]||_2 \leq \frac{1}{2\sqrt{\lambda_{\min}(A)}}||\text{vec}(A)||_2$ □

Let T and $\widehat{T}$ be defined as in Eq. (2.17) and Eq. (2.18).
The following Lemma holds for two general double stochastic matrices.

Lemma 2.3.6. *Let T and $\widehat{T}$ be two symmetric, stochastic matrices, it holds that :*

$$||T - \widehat{T}||_2 \leq \frac{\sqrt{C}||T^2 - \widehat{T}^2||}{\lambda_{\min}(T^2) - ||T^2 - \widehat{T}^2||_2} \quad \textit{and} \quad ||T - \widehat{T}||_2 \leq \frac{\sqrt{C}||T^2 - \widehat{T}^2||}{\lambda_{\min}(\widehat{T}^2) - ||T^2 - \widehat{T}^2||_2}$$

Proof. From the previous Lemma and the mean absolute value

$$||\sqrt{A} - \sqrt{B}||_2 \leq ||A - B||_2 \sup_{0\leq\theta\leq 1} ||D[\sqrt{\theta A + (1-\theta)B}]||_2$$

For Weyl's inequality

$$\begin{aligned}\lambda_{\min}(\theta T^2 + (1-\theta)\widehat{T}^2) &\leq \lambda_{\min}(\theta T^2) + \lambda_{\min}((1-\theta)\widehat{T}^2)\\ &= \theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)\end{aligned}$$

$$\begin{aligned}\sup_{0\leq\theta\leq 1} ||D\sqrt{\theta T^2 + (1-\theta)\widehat{T}^2}||_2 &\leq \frac{1}{2}\sup_{0\leq\theta\leq 1} \frac{||\text{vec}(\theta T^2) + (1-\theta)\widehat{T}^2)||_2}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)}\\ &\leq \frac{1}{2}\sup_{0\leq\theta\leq 1} \frac{\theta||\text{vec}(T^2)||_2 + (1-\theta)||\text{vec}(\widehat{T}^2)||_2}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)}\\ &\leq \frac{1}{2}\sup_{0\leq\theta\leq 1} \frac{||\text{vec}(T^2)||_2 + ||\text{vec}(\widehat{T}^2)||_2}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)}\\ &\leq \sup_{0\leq\theta\leq 1} \frac{\sqrt{C}}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)}\end{aligned}$$

In the last inequality we used that T and $\widehat{T}$ and doubly stochastic, meaning that $\sum_{i=1}^C T_{ij}^2 \leq (\sum_{i=1}^C T_{ij})^2 = 1$. So $||\text{vec}(T)||_2 = \left(\sum_{i=1}^C \sum_{j=1}^C T_{ij}^2\right)^{\frac{1}{2}} \leq \sqrt{C}$. Moreover deriving $\frac{1}{\theta\lambda_{\min}(T^2)+(1-\theta)\lambda_{\min}(\widehat{T}^2)}$ with respect to θ we find that

$$\sup_{0\leq\theta\leq 1} \frac{1}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)} = \begin{cases} \frac{1}{\lambda_{\min}(T^2)} & \text{if} \quad \lambda_{\min}(T^2) < \lambda_{\min}(\widehat{T}^2)\\ \frac{1}{\lambda_{\min}(\widehat{T}^2)} & \text{if} \quad \lambda_{\min}(T^2) > \lambda_{\min}(\widehat{T}^2)\end{cases}$$

$$\sup_{0\leq\theta\leq 1} \frac{1}{\theta\lambda_{\min}(T^2) + (1-\theta)\lambda_{\min}(\widehat{T}^2)} = \frac{1}{\min(\lambda_{\min}(\widehat{T}^2), \lambda_{\min}(T^2))}.$$

Now,

$$\min(a,b) = \begin{cases} a = b - |b-a| & \text{if } a < b\\ b & \text{if } b \leq a\end{cases} \tag{2.30}$$

We notice that for symmetric matrices $||A||_2 = \sqrt{\lambda_{\max}(A)^2} = \sqrt{(\lambda_{\max}(A))^2} = |\lambda_{\max}(A)|$. So we can Since $T^2 - \widehat{T}^2$ is symmetric: $||T^2 - \widehat{T}^2||_2 = |\lambda\max(T^2 - \widehat{T}^2)|$.

It follows that

$$\begin{aligned}\min(\lambda_{\min}(\widehat{T}^2), \lambda_{\min}(T^2)) &\geq \lambda_{\min}(T^2) - |\lambda_{\min}(T^2) - \lambda_{\min}(\widehat{T}^2)|\\ &\geq \lambda_{\min}(T^2) - |\lambda_{\min}(T^2) - \lambda_{\min}(\widehat{T}^2)|\\ &\geq \lambda_{\min}(T^2) - |\lambda_{\min}(T^2 - \widehat{T}^2)|\\ &\geq \lambda_{\min}(T^2) - |\lambda_{\max}(T^2 - \widehat{T}^2)|\\ &= \lambda_{\min}(T^2) - ||T^2 - \widehat{T}^2||_2.\end{aligned} \tag{2.31}$$

In the previous equations, we use that

$$|\lambda_{\min}(T^2) - \lambda_{\min}(\widehat{T}^2)| \leq |\lambda_{\max}(T^2 - \widehat{T}^2)|$$

. We now prove that it is true. Suppose without loss of generality that

$$\lambda_{\min}(T^2) > \lambda_{\min}(\widehat{T}^2)$$

. If it is the case

$$\lambda_{\min}(T^2) - \lambda_{\min}(\widehat{T}^2) = \lambda_{\min}(T^2) + \lambda_{\max}(-\widehat{T}^2) \leq \lambda_{\max}(T^2 - \widehat{T}^2) \leq |\lambda_{\max}(T^2 - \widehat{T}^2)|$$

, where we used Weyl's inequality.

If the $\lambda_{\min}(T^2) > \lambda_{\min}(\widehat{T}^2)$ following the same path we obtain

$$\lambda_{\min}(\widehat{T}^2) - \lambda_{\min}(T^2)| \leq |\lambda_{\max}(\widehat{T}^2 - T)|.$$

it follow that $\lambda_{\min}(T^2) - \lambda_{\min}(\widehat{T}^2) < ||T^2 - \widehat{T}^2||_2$

□

Proof Theorem Theorem 2.3.4. From Lemma 2.3.6 we know that

$$||T - \widehat{T}||_2 \leq \frac{\sqrt{C}||T^2 - \widehat{T}^2||}{\lambda_{\min}(T^2) - ||T^2 - \widehat{T}^2||_2} \tag{2.32}$$

Now, in general

$$\frac{\sqrt{C}x}{b - x} < \epsilon \text{ iif } x < b\frac{\epsilon}{\sqrt{C} + \epsilon}.$$

It follows that

$$\mathbb{P}(||T - \widehat{T}||_2 < \epsilon) = \mathbb{P}\left(||T^2 - \widehat{T}^2||_2 < \lambda_{\min}(T^2)\frac{\epsilon}{\sqrt{C} + \epsilon}\right)$$

or

$$\begin{aligned}\mathbb{P}(||T - \widehat{T}||_2 < \epsilon) &= \mathbb{P}(||T^2 - \widehat{T}^2||_2 < \lambda_{\min}(\widehat{T}^2)\frac{\epsilon}{\sqrt{C} + \epsilon})\\ &\geq \mathbb{P}(||T^2 - \widehat{T}^2||_2 < \frac{\lambda_{\min}(\widehat{T}^2)}{\sqrt{C} + 1}\epsilon)\end{aligned}$$

Since we can assume $\epsilon \leq 1$ (if $n > \frac{C^2(\sqrt{C}+1)^2(\ln(2C^2)^2}{2\lambda_{\min}(\widehat{T})^2}$. Notice that we are interested in convergence properties of $\widehat{T}$, so we are interested in founding these bounds for small ϵ.

Now $T^2 - \widehat{T}^2 = D^{1/2}(M - \widehat{M})D^{1/2}$.

So $||T^2 - \widehat{T}^2||_2 \leq ||M - \widehat{M}||_2||D^{1/2}||_2^2 = ||M - \widehat{M}||_2||D||_2 = ||M - \widehat{M}||_2\lambda_{\max}(D)$. As a consequence :

$$\begin{aligned}\mathbb{P}(||T - \widehat{T}||_2 < \epsilon) &\geq \mathbb{P}\left(||M - \widehat{M}||_2\lambda_{\max}(D) < \frac{\lambda_{\min}(\widehat{T}^2)}{\sqrt{C} + 1}\epsilon\right)\\ &= \mathbb{P}\left(||M - \widehat{M}||_2 < \frac{\lambda_{\min}(\widehat{T}^2)}{(\sqrt{C} + 1)\lambda_{\max}(D)}\epsilon\right)\\ &\geq 1 - 2C^2 e^{-\frac{\epsilon^2}{C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T}^2)^2}{\lambda_{\max}(D)^2}n}\end{aligned}$$

For the inverse:
$$T^{-1}-\widehat{T}^{-1}=T^{-1}(\widehat{T}-T)\widehat{T}^{-1}$$
So,
$$||T^{-1}-\widehat{T}^{-1}||_2 \le ||T^{-1}||_2||\widehat{T}-T||_2||\widehat{T}^{-1}||_2 = \frac{1}{\lambda_{\min}(T)\lambda_{\min}(\widehat{T})}||\widehat{T}-T||_2$$

Following what we did for the κ in

$$\frac{1}{\lambda_{\min}(T)\lambda_{\min}(\widehat{T})} \le \frac{1}{\min(\lambda_{\min}(T^2),\lambda_{\min}(\widehat{T}^2))} \le \frac{1}{\lambda_{\min}(\widehat{T}^2)-|\lambda_{\min}(T^2)-\lambda_{\min}(\widehat{T}^2)|}$$

Than for Eq. (2.31)
$$\frac{1}{\lambda_{\min}(T)\lambda_{\min}(\widehat{T})} \le \frac{1}{\lambda_{\min}(\widehat{T}^2)-||T^2-\widehat{T}^2||_2}$$

So
$$||T^{-1}-\widehat{T}^{-1}||_2 \le \frac{||T-\widehat{T}||_2}{\lambda_{\min}(\widehat{T}^2)-||T^2-\widehat{T}^2||_2} \le \frac{||T-\widehat{T}||_2}{\lambda_{\min}(\widehat{T}^2)-2||T-\widehat{T}||_2}$$

Where we used that
$$||T^2-\widehat{T}^2||_2 \le ||T(T-\widehat{T})+(T-\widehat{T})\widehat{T}||_2 \le 2||T-\widehat{T}||_2$$
because T and $\widehat{T}$ doubly stochastic.

So
$$\begin{aligned}\mathbb{P}\left(||T^{-1}-\widehat{T}^{-1}||_2 \le \epsilon\right) &\ge \mathbb{P}\left(||T-\widehat{T}||_2 \le \epsilon\frac{\lambda_{\min}(\widehat{T})}{1+2\epsilon}\right)\\ &\ge \mathbb{P}\left(||T-\widehat{T}||_2 \le \frac{\epsilon}{3}\lambda_{\min}(\widehat{T})\right)\\ &\ge 1-2C^2e^{-\frac{\epsilon^2}{9C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T}^2)^4}{\lambda_{\max}(D)^2}n}\end{aligned}$$

$\square$

From the previous theorem we can notice that the error in estimation of T decays as $\frac{1}{\sqrt{n}}$ as a function of n.

2.3.3 Learning from noisy labels

In this section, we show how to leverage the estimates of the error rates to train the models.

Posterior distribution of true labels as soft-labels

It is noteworthy that if we have access to the labels provided by all annotators, the posterior probabilities of the true labels can be calculated leveraging T and Bayes' Theorem as follows:

$$\underbrace{\mathbb{P}(y_i = c|y_{1,i}, \dots, y_{H,i})}_{:=p_{c,i}} \propto \nu_c \prod_{h=1}^{H} \underbrace{\mathbb{P}(y_{h,i}|y_i = c)}_{=T_{c,y_{h,i}}} \tag{2.33}$$

we recall that $\nu_c = \mathbb{P}(y_i = c)$ and that the conditional probabilities on the r.h.s. are given by T. In our case, we can use our noisy transition estimates to estimate the posterior probabilities of the true labels, and afterwards, we can use these posteriors to train the classifier.

Lemma 2.3.7. *For infinite annotators, the posterior distribution over every sample calculated using the true T converges to the Dirac delta distribution centred on the true label almost surely (i.e. $\lim_{H\to\infty} p_{c,i} \overset{a.s.}{=} \mathbb{1}(y_i = c)$).*

Lemma 2.3.8. *Let us consider a vector $\mathbf{x} = (x_1, \dots, x_n)$ s.t. $\sum_{i=1}^{n} x_i = 1$ and $x_i > 0$ for all i, and a vector $\mathbf{a} = (a_1, \dots, a_n)$ s.t $a_i > 0$ for $i = 1, \dots, n$. Let $\psi_{\mathbf{a}}(\mathbf{x}) = \prod_{j=1}^{n} x_j^{a_j}$, it holds that*

$$\underset{(x_i,\dots,x_n):\sum_i x_i = 1}{\operatorname{argmax}} \quad \psi_{\mathbf{a}}(\mathbf{x}) = (a_1, \dots, a_n)$$

Proof. Let us consider $\phi_{\mathbf{a}}(\mathbf{x}) = \log \psi_{\mathbf{a}}(\mathbf{x}) = \sum_{i=1}^{n} a_i \log(x_i)$. Recalling that $x_n = 1 - sum_{i=1}^{n-1} x_i$

$$\nabla\phi(\mathbf{x}) = \begin{bmatrix} \frac{a_1}{x_1} - \frac{a_n}{1-\sum_{i=1}^{n-1} x_i} \\ \frac{a_2}{x_1} - \frac{a_n}{1-\sum_{i=1}^{n-1} x_i} \\ \vdots \\ \frac{a_1}{x_{n-1}} - \frac{a_n}{1-\sum_{i=1}^{n-1} x_i} \end{bmatrix}$$

$$\nabla\phi(\mathbf{x}) \geq 0 \iff a_i(1 - \sum_{i=1}^{n-1} x_i) - a_n x_i \geq 0 \text{ for} i = 1, \dots, n-1$$

Namely, the maximum is reached for $\mathbf{x}$ that solves the following linear system has to be solved:

$$\begin{bmatrix} a_1 + a_n & a_1 & \dots & a_1 \\ a_2 & a_2 + a_n & \dots & a_2 \\ & \vdots & \ddots & \\ a_{n-1} & \dots & & a_{n-1} + a_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix}$$

We have that

$$A := \begin{bmatrix} a_1 + a_n & a_1 & \dots & a_1 \\ a_2 & a_2 + a_n & \dots & a_2 \\ & \vdots & \ddots & \\ a_{n-1} & \dots & & a_{n-1} + a_n \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \cdot \left[a_1, a_2, \dots, a_{n-1}\right] + a_n \mathrm{I}_{n-1}$$

A can be written as the sum of a rank-1 matrix and a_n times the identity. It holds that is $\sum_{i=1}^{n-1} a_i + a_n \neq 0$ then A is invertible so $\operatorname{rank}(A) = n - 1$ (Sherman and Morrison, 1949). The non-homogeneous system also has one unique solution. We know that $x_i = a_i$ is a solution, so it's the unique solution.

□

Proof of Lemma 2.3.7.

$$p_{c,i} = \frac{\mu_c \prod_{h=1}^{H} T_{c,y_{h,i}}}{\sum_{j=1}^{C} \mu_j \prod_{h=1}^{H} T_{j,y_{h,i}}} \tag{2.34}$$

$$\prod_{h=1}^{H} T_{c,y_{h,i}} = \prod_{j=1}^{C} T_{c,j}^{N_{i,j}} \tag{2.35}$$

where $N_{i,j}$ is the amount of annotators that labeled sample i as class j. Note that as a consequence of the strong law of large numbers for the conditional random variables that are independent with the same conditional distribution we have that the following equation is true almost surely:

$$\lim_{H\to\infty} \frac{N_{i,j}}{H} = \lim_{H\to\infty} \frac{\sum_{a=1}^{H} \mathbb{1}_{\{y_{a,i}=j\}}}{H} = \mathbb{E}[\mathbb{1}_{\{y_{a,i}=j\}}|y=j] = T_{y_i,j} \tag{2.36}$$

Combining we get:

$$\lim_{H\to\infty} p_{c,i} = \lim_{H\to\infty} \frac{\mu_c \prod_{j=1}^{C} T_{c,j}^{N_{i,j}}}{\sum_{k=1}^{C} \mu_k \prod_{j=1}^{C} T_{k,j}^{N_{i,j}}} \tag{2.37}$$

$$= \lim_{H\to\infty} \frac{\mu_c \left(\prod_{j=1}^{C} T_{c,j}^{T_{y_i,j}}\right)^H}{\sum_{k=1}^{C} \mu_k \left(\prod_{j=1}^{C} T_{k,j}^{T_{y_i,j}}\right)^H} \tag{2.38}$$

$$= \lim_{H\to\infty} \frac{1}{1 + \sum_{\substack{k=1\\k\neq c}}^{C} \frac{\mu_k}{\mu_c} \left(\prod_{j=1}^{C} \left(\frac{T_{k,j}}{T_{c,j}}\right)^{T_{y_i,j}}\right)^H} \tag{2.39}$$

$$\overset{(a)}{=} \mathbb{1}(y_i = c) \tag{2.40}$$

where in (a) we used the fact that due to the assumption that T is strictly dominant, then the term $\prod_{j=1}^{C} T_{k,j}^{T_{y_i,j}}$ is maximized when $k = y_i$ and this term is strictly larger than all the other ones, see Lemma 2.3.8. Indeed, if there exists k s.t. $\prod_{j=1}^{C} \left(\frac{T_{k,j}}{T_{c,j}}\right)^{T_{y_i,j}} > 1$ than $\lim_{H\to infty} p_{c,i} = 0$ because the denominator goes to ∞.

So the only case for not having is that $\prod_{j=1}^{C} \left(\frac{T_{k,j}}{T_{c,j}}\right)^{T_{y_i,j}} \leq 1$ for all k. Suppose we know that the maximum of the function $\prod_{j=1}^{C} (x_i)^{a_i}$ is reached for $x_i = a_i \; \forall i = 1, \ldots, C$ than we're done.

Indeed, we have that $\prod_{j=1}^{C} \left(\frac{T_{k,j}}{T_{c,j}}\right)^{T_{y_i,j}} > 1$, if and only if $\prod_{j=1}^{C} (T_{k,j})^{T_{y_i,j}} > \prod_{j=1}^{C} (T_{c,j})^{T_{y_i,j}}$ since we're considering all k, for $k = 1, \ldots, C$, $k \neq c$. If $y_i \neq c$ it means that y_i is one of the values k can assume and since that one is the max, it means that for sure it will be greater than $\prod_{j=1}^{C} (T_{c,j})^{T_{y_i,j}}$.

Otherwise, if $y_i = c$ it means that $\prod_{j=1}^{C} (T_{c,j})^{T_{y_i,j}} > \prod_{j=1}^{C} (T_{k,j})^{T_{y_i,j}}$ so all elements are less than 1 and the limit goes to 1. □

We can use the posterior distributions as soft-labels defining the following loss for the i-th sample:

$$\ell(f(x_i), y_{1,i}, \dots, y_{H,i}) = \ell(f(x_i), \bar{p}_i) \tag{2.41}$$

where $\bar{p}_i = [p_{1,i}, \cdots, p_{C,i}]^T$. Or we can use the posterior distributions to weight the loss function at the i-th sample evaluated at each of the possible labels:

$$\ell(f(x_i), y_{1,i}, \dots, y_{H,i}) = \sum_{c=1}^{C} p_{c,i} \ell(f(x_i), e_c) \tag{2.42}$$

where e_c is the vector in $\mathbb{R}^C$ with 1 in the c-th position. Notice that for categorical cross-entropy loss, the two functions defined above correspond, but in general, they define two different loss functions.

Note that these soft labels are different from the ones obtained by averaging the annotator's labels as is done in Wei et al. (2022). The method using the posteriors exploits the T matrix and thus more information than the simple mean of the values of the losses among annotators. We, therefore, expect this to yield better results than the aggregation using the mean proposed in Wei et al. (2022). These considerations are supported by the empirical results we obtained on synthetic datasets (see Sec. 2.5).

Robust loss functions

Another way to leverage the estimate of T is to use robust loss functions, like the forward and backward loss functions presented in Natarajan et al. (2013) and Patrini et al. (2017). Let $\ell(t, y)$ be a generic loss function for the classification task, with a little abuse of notation we define $\ell(t) = [\ell(t, e_1), \dots, \ell(t, e_C)]^T$. The backward and forward loss functions are defined in Eq. (2.43a) and Eq. (2.43b), respectively.

$$l_b(t, y) = (\widehat{\Gamma}^{-1} \ell(t)) y \tag{2.43a}$$

$$l_f(t, y) = (\ell(\widehat{\Gamma}^T t)) y \tag{2.43b}$$

To explain the notation in Eq. (2.43a) we are first doing the dot product between the matrix Γ^{-1} and the vector $\ell(t)$ and then the dot product of the resulting vector with y. These losses leverage aggregated labels and therefore different aggregating techniques can be used, like majority vote. Another possible aggregating technique that leverages the posterior probabilities is to assume that the true label is the one that corresponds to the class that has the highest posterior probability.

2.3.4 Generalizations gap bounds

In this section, we derive generalization gap bounds for the backward loss that depends on the noise transition matrix estimated in Eq. (2.19). Since we are only addressing the problem for the backward loss, from now on we will denotethe backward loss by l.

Remark 2. *If $\ell(t, y)$ is Lipschitz with constant L, the loss function $l(t, y)$ is Lipschitz with Lipschitz constant $||\Gamma^{-1}||_2 L$.*

We will prove the following theorem in the case of $\Gamma = T$. We emphasize that all the results apply also when $\Gamma^{-1} = \phi(T^{-1})$ and that the function that associate Γ^{-1} and T^{-1} ,ϕ is Lipschitz with respect to the norm p, i.e. there exists a Lipschitz constant $L_{\phi,p}$ s.t. $||\phi(T^{-1}) - \phi(\widehat{T}^{-1})||_p \leq L_{\phi,p}||T^{-1} - \widehat{T}^{-1}||_p$. The only difference is that in the bound we will have a factor $L_{\phi,p}$.

It has been proved, first in Natarajan et al. (2013) (Lemma 1) for the binary classification task and then in general for the multi-class case in Patrini et al. (2017) (Theorem 1) that $l(t, y)$ is an unbiased estimator for ℓ, i.e.

$$\mathbb{E}_{\tilde{y}|y}[l(t, \tilde{y})] = \ell(t, y).$$

Lemma 2.3.9. *Let ℓ be a bounded loss function, so that the image of ℓ is in $[0, \mu]$, and s.t. ℓ is Lipschitz in the first argument with Lipschitz constant L. Let $\widehat{R}_l(f)$ be the empirical risk for the loss l and let $R_{l,\mathcal{D}}$ be the risk for a loss l under the distribution $\mathcal{D}$, with l unbiased estimator for the loss ℓ. We denote by $\hat{l}$ the backward loss obtained using $\widehat{T}$.*

$$\sup_{f\in\mathcal{F}} |\hat{R}_{\hat{l}}(f) - R_{l,\mathcal{D}}(f)| \leq \left[L\lambda_{\min}(\widehat{T}^2) + \frac{\mu\lambda_{\min}(D)}{\lambda_{\min}(\widehat{T})^2}\sqrt{\frac{1}{n}\ln\left(\frac{4C}{\delta}\right)}\right]\mathfrak{R}_n(\mathcal{F})g(C).$$

with $g(C) = 6C^2(\sqrt{C} + 1)$

Proof. For every f we have

$$|\hat{R}_{\hat{l}}(f) - R_{l,\mathcal{D}}(f)| \leq |\hat{R}_{\hat{l}}(f) - \hat{R}_l(f)| + |\hat{R}_l(f) - R_{l,\mathcal{D}}(f)|.$$

So using union bounds and by the classic results on Rademacher complexity bounds (Mohri et al., 2012), and by the Lipschitz composition property of Rademacher averages, Theorem 7 in Meir and Zhang (2003) it follows that

$$\begin{aligned}
&\mathbb{P}^n\Big(\sup_{f\in\mathcal{F}} |\hat{R}_{\hat{l}}(f) - R_{l,\mathcal{D}}(f)| \leq L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F}) + \frac{\epsilon}{2}\Big) \\
&\geq \mathbb{P}^n\Big(\sup_{f\in\mathcal{F}} |\hat{R}_{\hat{l}}(f) - \hat{R}_l(f)| + \sup_{f\in\mathcal{F}} |\hat{R}_l(f) - R_{l,\mathcal{D}}(f)| \leq L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F}) + \frac{\epsilon}{2}\Big) \\
&\geq 1 - \mathbb{P}^n\Big(\sup_{f\in\mathcal{F}} |\hat{R}_{\hat{l}}(f) - \hat{R}_l(f)| > \frac{\epsilon}{4}\Big) - \mathbb{P}^n\Big(\sup_{f\in\mathcal{F}} |\hat{R}_l(f) - R_{l,\mathcal{D}}(f)| \leq L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F}) + \frac{\epsilon}{4}\Big) \\
&\geq 1 - \mathbb{P}^n\Big(\sup_{f\in\mathcal{F}} |\hat{R}_{\hat{l}}(f) - \hat{R}_l(f)| > \frac{\epsilon}{4}\Big) - 2e^{-\frac{n}{2}\left(\frac{\epsilon}{4\mu}\right)^2}
\end{aligned}$$

Now,

$$
\begin{aligned}
&\mathbb{P}^n\Big(\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-R_{l,\mathcal{D}}(f)|\le L||\widehat{T^{-1}}||_2\mathfrak{R}_n(\mathcal{F})+\epsilon\Big)\\
&=\mathbb{P}^n\Big(\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-R_{l,\mathcal{D}}(f)|\le L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F})+(||\widehat{T^{-1}}||_2-||T^{-1}||_2)\mathfrak{R}_n(\mathcal{F})+\epsilon\Big)\\
&\ge 1-\mathbb{P}^n\Bigg(\{\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-R_{l,\mathcal{D}}(f)|\le L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F})+\frac{\epsilon}{2}\}\\
&\qquad\qquad\cap\{(||\widehat{T^{-1}}||_2-||T^{-1}||_2)\mathfrak{R}_n(\mathcal{F})\le\frac{\epsilon}{2}\}\Bigg)\\
&\ge 1-\mathbb{P}^n\Big(\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-R_{l,\mathcal{D}}(f)|\le L||T^{-1}||_2\mathfrak{R}_n(\mathcal{F})+\frac{\epsilon}{2}\Big)\\
&\quad-\mathbb{P}^n\Big((||\widehat{T^{-1}}||_2-||T^{-1}||_2)\mathfrak{R}_n(\mathcal{F})\le\frac{\epsilon}{2}\Big)\\
&\ge 1-2e^{-\frac{n}{2}\left(\frac{\epsilon}{4\mu}\right)^2}-\mathbb{P}^n\Big((||\widehat{T^{-1}}-T^{-1}||_2)\le\frac{\epsilon}{2\mathfrak{R}_n(\mathcal{F})}\Big)-\mathbb{P}^n\Big(\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-\hat{R}_l(f)|>\frac{\epsilon}{4}\Big)\\
&\ge 1-2e^{-\frac{n}{2}\left(\frac{\epsilon}{4\mu}\right)^2}-2C^2e^{-\frac{\epsilon^2}{4\mathfrak{R}_n(\mathcal{F})^2 9C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{\lambda_{\max}(D)^2}n}-2C^2e^{-\frac{\epsilon^2}{4\mu^2 9C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{\lambda_{\max}(D)^2}n}\\
&\ge 1-2e^{-\frac{n}{2}\left(\frac{\epsilon}{4\mu}\right)^2}-4C^2e^{-\frac{1}{\max(\mathfrak{R}_n(\mathcal{F}),\mu)^2}\frac{\epsilon^2}{36C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{\lambda_{\max}(D)^2}n}\\
&\ge 1-4e^{-\left[\min\left(\frac{1}{8},2\ln(C)\frac{1}{9\mathfrak{R}_n(\mathcal{F})^2C^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{(\sqrt{C}+1)^2\lambda_{\max}(D)^2}\right)\right]\frac{\epsilon^2}{4\mu^2}n}\\
&\ge 1-4Ce^{-\left(\frac{1}{9\mathfrak{R}_n(\mathcal{F})^2C^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{(\sqrt{C}+1)^2\lambda_{\max}(D)^2}\right)\frac{\epsilon^2}{2\mu^2}n}
\end{aligned}
$$

So with probability at least $1-\delta$

$$\sup_{f\in\mathcal{F}}|\hat{R}_{\hat{l}}(f)-R_{l,\mathcal{D}}(f)|\le\left[2L\lambda_{\min}(\widehat{T}^2)+\frac{\mu\lambda_{\min}(D)}{\lambda_{\min}(\widehat{T})^2}\sqrt{\frac{1}{n}\ln\left(\frac{4C}{\delta}\right)}\right]\mathfrak{R}_n(\mathcal{F})g(C).$$

with $g(C)=6C^2(\sqrt{C}+1)$

$\square$

Theorem 2.3.10. *Let l be an unbiased estimator for ℓ defined as in Eq. (2.43a), Denoting $\hat{f}=\underset{f}{\operatorname{argmin}}(\widehat{R}_{\hat{l}}(f))$. It holds that*

$$R_{\ell,\mathcal{D}}(\hat{f})-\min_{f\in\mathcal{F}}R_{\ell,\mathcal{D}}(f)\le\left[2L\lambda_{\min}(\widehat{T}^2)+\frac{\mu\lambda_{\min}(D)}{\lambda_{\min}(\widehat{T})^2}\sqrt{\frac{1}{n}\ln\left(\frac{4C}{\delta}\right)}\right]\mathfrak{R}_n(\mathcal{F})g(C)$$

with $g(C)=6C^2(\sqrt{C}+1)$

In order to prove the previous theorem we need the following Proposition.

Proposition 2.3.10.1. *Let $\ell(t,y)$ be any bounded loss function and let $l(t,y)$ be the backward loss function defined in Eq. (2.43a).*

We define $\hat{l}(t,y)$ as the loss obtained using $\widehat{\Gamma}^{-1} := \widehat{T}^{-1}$. If μ is the constant that bounded the loss ℓ , i.e. $\sup_{(t,y)\in[0,1]^C\times\mathcal{Y}} \ell(t,y) \leq \mu$. *For every ϵ*

$$\mathbb{P}(|l(t,y) - \hat{l}(t,y)| \geq \epsilon) \leq 2C^2 e^{-2\frac{\epsilon^2}{C^2\mu^2 L_{\phi,p}}n} \tag{2.44}$$

Proof of Proposition 2.3.10.1. Using Cauchy–Schwarz inequality and the fact that ℓ is bounded by μ and that we obtain:

$$\begin{aligned}
|l(t,y) - \hat{l}(t,y)| &= |(T^{-1}\cdot\ell(t) - \widehat{T}^{-1}\cdot\ell(t))_y| \\
&= |[(T^{-1} - \widehat{T}^{-1})\ell(t)]\cdot e_y| \\
&\leq ||(T^{-1} - \widehat{T}^{-1})\ell(t)||_2||e_y||_2 \\
&\leq ||T^{-1} - \widehat{T}^{-1}||_2||\ell(t)||_2 \\
&\leq \mu||T^{-1} - \widehat{T}^{-1}||_2
\end{aligned}$$

So

$$\mathbb{P}(|l(t,y) - \hat{l}(t,y)| \leq \epsilon) \geq 1 - 2C^2 e^{-\frac{\epsilon^2}{\mu^2 9C^2(\sqrt{C}+1)^2}\frac{\lambda_{\min}(\widehat{T^2})^4}{\lambda_{\max}(D)^2}n}$$

□

Proof of Theorem 2.3.10 . By the unbiasedness of l we have that $R_{\ell,\mathcal{D}}(\hat{f}) = R_{l,\mathcal{D}}(\hat{f})$. Moreover since $\hat{f} = \underset{f}{\operatorname{argmin}}(\widehat{R}_{\hat{l}}(f))$ we have $\widehat{R}_{\hat{l}}(\hat{f}) \leq \widehat{R}_{\hat{l}}(g)\ \forall g \in \mathcal{F}$.

Let f^* be so that $\min_{f\in\mathcal{F}} R_{\ell,\mathcal{D}}(f) = R_{\ell,\mathcal{D}}(f^*)$. It follows that

$$\begin{aligned}
R_{\ell,\mathcal{D}}(\hat{f}) - \min_{f\in\mathcal{F}} R_{\ell,\mathcal{D}}(f) &= R_{l,\mathcal{D}}(\hat{f}) - \min_{f\in\mathcal{F}} R_{l,\mathcal{D}}(f) \\
&= R_{l,\mathcal{D}}(\hat{f}) - \widehat{R}_{l,\mathcal{D}}(\hat{f}) + \widehat{R}_{l,\mathcal{D}}(\hat{f}) - R_{\ell,\mathcal{D}}(f^*) \\
&\geq R_{l,\mathcal{D}}(\hat{f}) - \widehat{R}_{\hat{l},\mathcal{D}}(\hat{f}) - (R_{\ell,\mathcal{D}}(f^*) - \widehat{R}_{\hat{l},\mathcal{D}}(f^*)) \\
&\geq 2\max_{f\in\mathcal{F}} |R_{\ell,\mathcal{D}}(f) - \widehat{R}_{\hat{l},\mathcal{D}}(f)|
\end{aligned}$$

□

We observe that in all the previous theorems, the bounds found are always decreasing as one over the square root of the number of samples. The above theorem gives us a performance bound for the classifier found minimizing the backward loss l, i.e. the unbiased estimator of the loss ℓ on the noisy dataset. The bounds found depend on, the Rademacher complexity of the function space and the Lipschitz constant of the loss function.The importance of these bounds lies in the fact that they allow us to obtain performance bounds for a model trained with noisy data that depends on values that we can estimate from the noisy dataset. In particular, there is no dependence on the true noise transition matrix of the annotators, as in other work (Natarajan et al., 2013) which is instead a quantity that cannot be known a priori having access only to the training data. More in detail the bound depends on the estimate noise transition matrix, the number of classes in the dataset, the Rademacher complexity and the Lipschitz constant, which we can take as known a priori and on the distribution of ground truth, which in many cases it makes sense to assume uniform.

2.4 Cohen's kappa. Symmetric noise and symmetric ground truth distribution

We can also consider the case where an estimate of the IAA matrix M is not available and we only have access to a scalar representation of the inter-annotator agreement like Cohen's κ. Cohen's κ coefficient measures the agreement between two raters who each classify n items into C mutually exclusive categories.

We define the agreement among raters a and b as p_o: $p_o = \sum_{c=1}^{C} \mathbb{P}(y_a = c \cap y_b = c)$ Cohen and others (Cohen, 1960) suggest comparing the actual agreement (p_o) with the "chance agreement" that could be obtained if the labels assigned by the two annotators were independent (we will denote this quantity by p_e).

$$p_e = \sum_{c=1}^{C} \mathbb{P}(y_a = c)\mathbb{P}(y_b = c) \tag{2.45}$$

Cohen's κ coefficient is defined as the difference between the true agreement and the "chance agreement" normalized by the maximum value this difference can reach

$$\kappa := \frac{p_o - p_e}{1 - p_e}, \tag{2.46}$$

If the raters are in complete agreement then $\kappa = 1$. If there is no agreement among the raters other than what would be expected by chance (i.e. $p_o = p_e$) $\kappa = 0$. It can also take negative values. A negative κ indicates an agreement worse than that expected by chance. This can be interpreted as no agreement at all between annotators. In our work, we assume that the two raters are a corrupted version of an observable "clean" (ground truth) label. In this setting, the label assigned by annotator a to an item and the respective uncorrupted label are not independent random variables. We found that in this setting the κ coefficient can take only non-negative values.

In the case of symmetric noise and symmetric ground truth distribution, we need to estimate just one parameter and hence the matrix T has to be parameterized by a single parameter that can be estimated.

One particular example is the case where the noise is uniform among classes. Under these hypotheses, T is a matrix with all values $1-p$ on the diagonal and $\frac{p}{C-1}$ off the diagonal.

Lemma 2.4.1 (Relationship between p and κ)**.** *In the case of classification with uniform noise for two homogeneous annotators with noise rate p, i.e if a is one annotator, $\mathbb{P}(y_a = i | y = j) = p$ if $i \neq j$. If the distribution of the ground-truth labels is uniform, it holds that:*

$$p = (1 - C^{-1})(1 - \sqrt{\kappa}) \tag{2.47}$$

with κ the Cohen's kappa coefficient of the two annotators.

Proof.

$$
\begin{aligned}
p_o = \mathbb{P}(y_a = y_B) &= \sum_{k,h=1}^{C} \mathbb{P}(y_A = k, y_B = k | y = h)\mathbb{P}(y = h) \\
&= \sum_{k,h=1}^{C} \mathbb{P}(y_A = k | y = h)\mathbb{P}(y_B = k | y = h)\nu_h = \sum_{k,h=1}^{C} T_{h,k}^2 \nu_h \\
&= \sum_{h=1}^{C} (1-p)^2 c_h + \sum_{h=1}^{C} \left(\frac{p}{C-1}\right)^2 (C-1)c_h = (1-p)^2 + \frac{p^2}{C-1}
\end{aligned}
$$

Now

$$
\mathbb{P}(y_B = k) = \sum_{h=1}^{C} \mathbb{P}(y_B = k | y = h)\mathbb{P}(y = h) = \sum_{h=1}^{C} T_{hk}\nu_h = (T\nu)_k
$$

In the previous equation, we used that T is symmetric.

$$
\begin{aligned}
p_e = \sum_{k=1}^{C} \mathbb{P}(y_A = k)\mathbb{P}(y_B = k) &= \sum_{k=1}^{C} \mathbb{P}(y_A = k)\mathbb{P}(y_B = k) = c^T T^2 c \\
&= 2\frac{p}{C-1} - \frac{Cp^2}{(C-1)^2} + \left(1 - \frac{Cp}{C-1}\right)^2 \nu^T \nu
\end{aligned} \tag{2.48}
$$

If the distribution of the true label y is symmetric the probability vector $\nu = (\frac{1}{C}, \ldots, \frac{1}{C})$ So $\nu^T \nu = \frac{1}{C}$ and so

$$
\kappa = \frac{C^2 p^2 - 2C(C-1)p + (C-1)^2}{(C-1)^2} \tag{2.49}
$$

From which it follows that

$$
p = (1 - C^{-1})(1 - \sqrt{\kappa}) \tag{2.50}
$$

□

If T is assumed to be of the form described above (with all diagonal elements equal to $1-p$ and all off-diagonal entries equal), it has one eigenvalue equal to 1 and all the rest are equal to $1 - pC(C-1)^{-1}$ (this follows from the fact that in this case T can be written as a weighted summation of the identity and a rank-one matrix). Hence using Eq. (2.47) we get that $\lambda_{\min}(T) = \sqrt{\kappa}$. The bounds from Theorem 2.3.10 holds replacing $\lambda_{\min}(T)$ with $\sqrt{\kappa}$. This allows us to obtain a bound for the generalization gap of a classifier trained with backward loss even in the case where a single statistic on the agreement between annotators is provided.

2.5 Experimental results

We performed experiments to validate the effectiveness of the method we propose for estimating $\widehat{T}$ by studying the error in the estimation as a function of the number of samples. We also performed experiments to show how the estimated T

can be leveraged to train classifiers in the presence of noise labels. In particular, we performed experiments for a classification task on a synthetic dataset and on the CIFAR10-N dataset, comparing the performance of a classifier trained using labels obtained by some baseline aggregation method with the performance of a classifier trained using the distribution of posteriors obtained from the estimation of T (Eq. (2.33)) as soft-labels.

Estimation of T With these experiments, we aim to validate the theoretical results of Sec. 2.3.1. The results were obtained from a synthetic, generated dataset in which we generate the classes predicted by the annotators according to various T matrices. For each annotator, we produce their prediction according to the matrix T. We run experiments for the number of annotators $H = 10, 7, 3, 2$. For experiments with $2, 3$ and 7 annotators, we generate T as all possible symmetric, stochastic and diagonally dominant matrices with $[0.1, 0.2, 0.3, 0.4, 0.5]$ out of the diagonal and $[0.6, 0.8, 1.0]$ on the diagonal. Classes are uniformly distributed. For experiments with 10 annotators, we generate the matrices T as all possible (admissible) combinations that have $[0, 0.2, 0.4]$ out of the diagonal and $[0.6, 0.8, 1.0]$ on the diagonal. In this case, we both include uniform distribution of the true labels among the 4 classes and all the distributions so that the four classes can be partitioned into two groups of indices so that classes in the same group have the same probability. Namely, if the distributions on the classes are given by $= [d_1, d_2, d_3, d_4]$, admissible distributions are the ones for which there are two subsets if indices I and J so that $I \cup J = \{0, 1, 2, 3, 4\}$ and for all $i, k \in I : d_i = d_k$. The probability of the classes takes value in $[0.1, 0.2, 0.3, 0.4]$. This means that, for instance, we will find the distribution $[0.3, 0.3, 0.3, 0.1]$ or the distribution $[0.4, 0.1, 0.1.0.4]$ but not $[0.3, 0.2, 0.1, 0.4]$. Results for $2, 3$ and 7 annotators were obtained by averaging over 3 runs. Results for 10 annotators were obtained by averaging over 10 runs. The error that appears on axis y in the plots is the difference in norm 2 of the true matrix T and the estimated matrix $\widehat{T}$, obtained as explained in Sec. 2.3.1.

We recall that if the minimum eigenvalue is 1, the matrix T is the identity, and thus, the annotators always predict the exact class. The smaller the minimum eigenvalue, the noisier the dataset will be. In Fig. 2.3 as well as in Fig. 2.2, we can observe that the error in the estimation decreases as $\frac{1}{\sqrt{n}}$ with n number of samples, which is in agreement with the bound provided in Theorem 2.3.4. The results with respect to the minimum eigenvectors and with respect to the maximum diagonal value are consistent with each other and very similar. We also observed that, as expected, the estimation becomes more accurate as the number of annotators increases.

We can notice in Fig. 2.2 that the estimation becomes more precise as the number of annotators increases.

With Fig. 2.3 we wanted to see if datasets with a higher noise level have higher approximation errors than less noisy datasets. The plots show a minor trend: the estimation error also decreases as the noise decreases. The trend is not particularly noticeable perhaps due to a large number of annotators.

We recall that if the minimum eigenvalue is 1 or the maximum value of the diagonals is 1, the matrix T is the identity, and thus, the annotators always predict the exact class.

The smaller the minimum eigenvalue or the maximum value on the diagonal, the noisier the dataset will be.

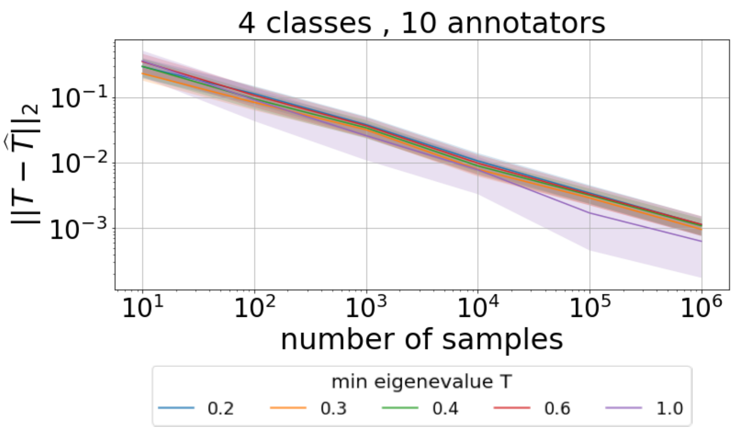


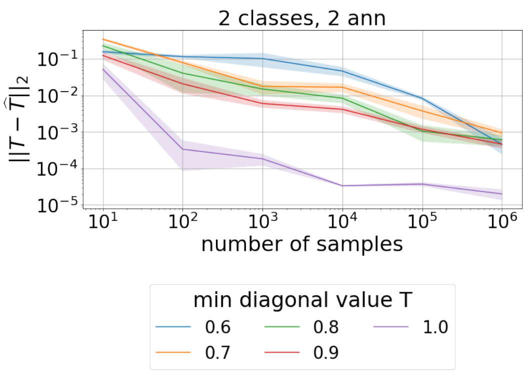
Figure 2.1. Error in the estimation of T for 4 classes and 10 annotators. The plots are obtained by averaging different admissible matrices T (see Sec. 2.3.2) and averaged over matrices that have the same minimum eigenvalues rounded to the first decimal.

Classification task with synthetic data We consider a classification task with a synthetic dataset. The features are generated uniformly in $[0,1]^2$. The assignment of labels (y) is done by following the label distribution established for each experiment, separating the space with lines parallel to the bisector of the first and third quadrants (specifically, $x_2 = x_1$). See Fig. 2.4 for an example. Our dataset comprises 10000 samples.

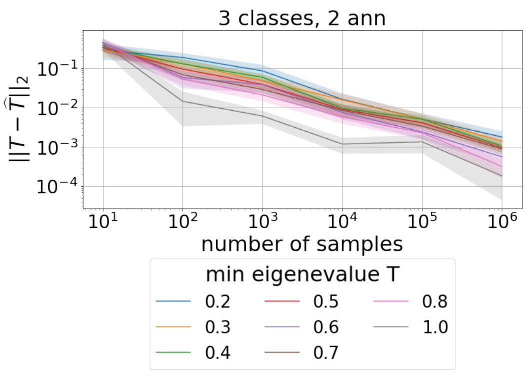
For each dataset, annotations are generated according to the noise transition matrix T. Various combinations of T are tested that respect the assumptions of symmetry, stochasticity and diagonally dominance, as well as being commutative with D (more details can be found in 2.3.2). The number of annotators is variable in the set $\{3, 5\}$.

Losses We use categorical cross entropy as loss function. We use both hard labels and soft labels to train the models.

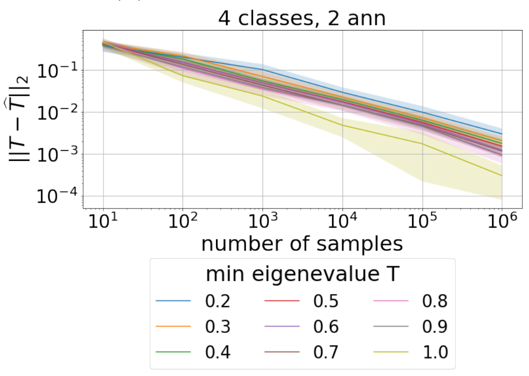
To train the models with hard labels an aggregation method is needed to obtain one final label from the annotators. We consider random and majority votes. In random aggregation, the final label is randomly picked from the labels of the annotators. In the majority vote the final label is the one with the most amount of votes (the mode), if the mode is not unique, we randomly choose one of the most voted classes. As soft labels, we consider the relative frequency among annotators and the posterior distribution according to Eq. (2.33). In the case of frequency for each sample we average the one-hot encoded annotations. Notice that random, majority vote and frequency soft labels do not leverage the estimate of T while the posterior does. In Fig. 2.5 we report the results for 4 classes with distribution $(0.4, 0.1, 0.4, 0.1)$ and 3 annotators. In more detail, Fig. 2.6 show the results of the different aggregation methods, for different amounts of noise, when using a neural network without hidden layer (i.e. a Logistic Regression) trained with Cross Entropy Loss. When noise is absent, we check that, as expected, the results are all identical. In the presence of noise (0.6 and 0.8), we notice in general that the random aggregation is the worst. The others are equivalent, except for the posterior (ours) which obtains slightly higher results. Average, on the other hand, obtains a slightly lower value with minimum diagonal value of T equal to 0.8. However, attention must be drawn to the fact that the y-scale of the graph is very narrow



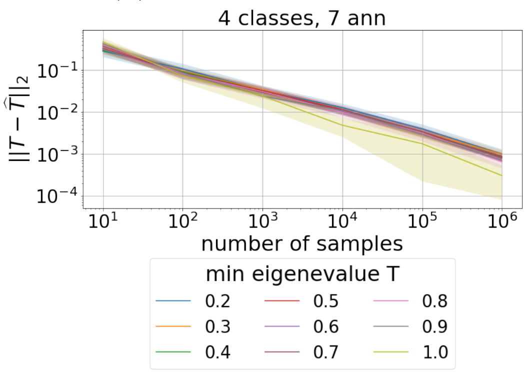
(a) 2 classes 2 annotators



(b) 3 classes 2 annotators

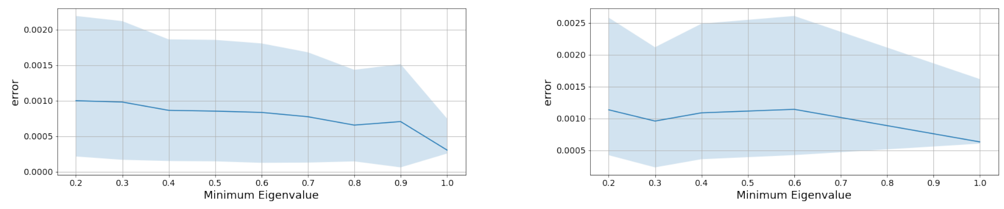


(c) 4 classes 2 annotators



(d) 4 classes 7 annotators

Figure 2.2. Error in the Estimation of T. The error is $||T - \widehat{T}||_2$. We aggregated the matrices with the same minimum eigenvalue rounded at the first decimal.



(a) 4 classes 7 annotators **(b)** 4 classes 10 annotators

Figure 2.3. The plots show the trend of the error estimation as the minimum eigenvalue increases

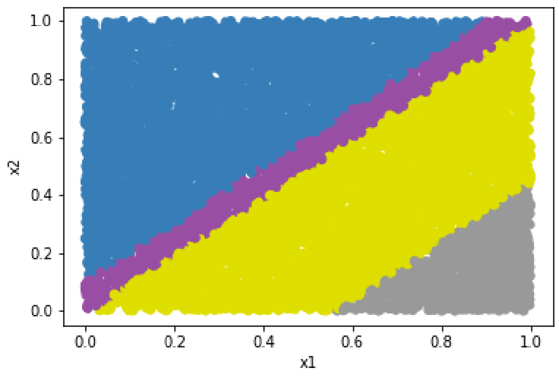


Figure 2.4. Synthetic data for 4 classes with distribution (0.4,0.1,0.4,0.1)

and that in the case of 4 classes with a dataset constructed as in Fig. 2.4, a linear classifier is not able to reach perfect accuracy.

We use accuracy with respect to a clean dataset as a performance metric.

Referring to Fig. 2.5 and the other figures of this section. The minimum value on the diagonal of the matrix T denotes the annotators' probability of assigning the correct label for the class in which the noise is maximum. As expected, random aggregation is the lowest performing method, and for all noise rates soft label methods perform better than methods using hard labels.

Fig. 2.6 shows the accuracy for the case of 4 classes and a NN with no hidden layer and 5 annotators. We can notice that even when the number of hidden neurons is insufficient to obtain perfect accuracy. Hence, the classifier is not the best possible; our approach for a high-noise dataset performs better.

The posteriors distribution is computed using the estimated T.

In conclusion, our results on this dataset show that using the posteriors distribution, as soft labels, allows for better performance than using the average of the labels assigned by annotators and then using majority vote or random aggregation. Our method is shown to be more robust to the noise and is also the one with less variance in the results. This confirms our hypotheses that by leveraging the matrix $\widehat{T}$ better classification accuracy can be achieved.

Experiments on CIFAR10-N The CIFAR10-N dataset[1] contains CIFAR-10 train images with noisy labels annotated by humans using Amazon Mechanical Turk. Each

[1]http://www.noisylabels.com

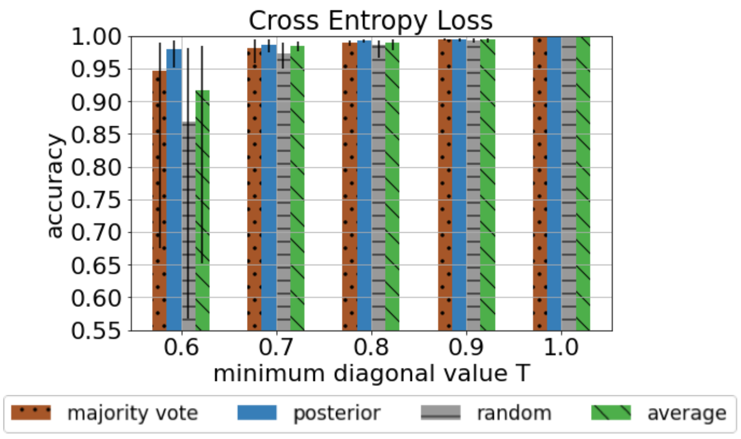


Figure 2.5. Comparison between the performance of Cross Entropy Loss using majority vote, random aggregation method or the posteriors (posterior) and relative frequency (average) as soft labels.On the y-axis the accuracy on a clean dataset and on the x-axis the values of the minimum on the diagonal of T. Small values of the minimum diagonal value mean a noisy dataset, while the minimum is 1 in the noise-free case. The results are obtained for 3 annotators and 4 classes, by averaging on different admissible matrices T (see Sec. 2.3.2) that have the same minimum diagonal values rounded to the first decimal. The error bands show the maximum and minimum performance for each method.

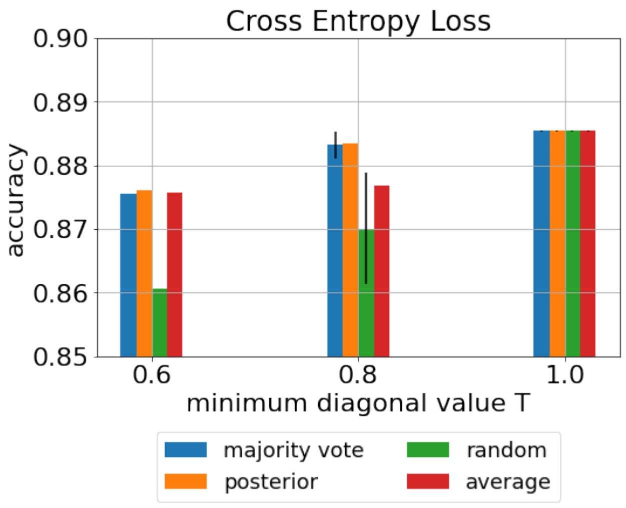


Figure 2.6. 5 annotators, 4 classes, no hidden layer.

Figure 2.7

image is labelled by three independent annotators. Table 2.1 shows the accuracy achieved using the different aggregation methods. For this experiment, we used Resnet34 (Khetan et al., 2017) with and without pre-training. In both cases, our approach of aggregation achieves the best performance. Note that in this dataset there are no guarantees that the assumptions we made on T are satisfied, however, the method is still applicable with positive results.

Aggregation Method	Pretrained	Not-Pretrained
random	0.718 ± 0.035	0.579 ± 0.023
majority vote	0.740 ± 0.017	0.590 ± 0.006
average	0.762 ± 0.012	0.637 ± 0.016
posteriors (ours)	$\mathbf{0.794 \pm 0.005}$	$\mathbf{0.652 \pm 0.014}$

Table 2.1. Test Accuracy on CIFAR10-N with Resnet34

2.6 Concluding remarks

We have addressed the problem of learning from noisy labels in the case where the dataset is labelled by annotators that occasionally make mistakes. We have introduced a methodology to estimate the noise transition matrix T of the annotators given the IAA. We further showed different techniques to leverage this estimate to learn from the noisy dataset in a robust manner. We have shown theoretically that the methods we introduce are sound. We supported our methodology with some experiments that confirm our estimation of the noise transition matrix is valid and that this can be leveraged in the learning process to obtain better performance.

Limitations The main limitation of our current approach to estimating T is that it only considers the case where T is symmetric and D assumed to be known and commutes with T. Extending the results to the case where T might not be symmetric and different among annotators is one possible future research direction.

Chapter 3

On Generalization Bounds for Clustering

Among the central questions in machine learning is, given a sample of n points P drawn from some unknown but fixed distribution $\mathcal{D}$, how well does a classifier trained on P generalize to $\mathcal{D}$? The probably most popular way to formalize this question is, given a loss function L and optimal solutions $\mathcal{S}_P$ and $\mathcal{S}_\mathcal{D}$ for sample P and distribution $\mathcal{D}$, respectively, how the empirical excess risk $L(\mathcal{D}, \mathcal{S}_P) - L(\mathcal{D}, \mathcal{S}_\mathcal{D})$ decreases as a function of n. This work focuses on loss functions associated with the clustering problem. Popular examples include (k, z) clustering, which asks for a set of k centers $\mathcal{S} \subset \mathbb{R}^d$ minimizing the cost $\sum_{p \in P} \min_{s \in \mathcal{S}} \|p - s\|_2^z$ and more generally, (k, j, z) subspace clustering which asks for a set of k subspaces $\mathcal{U} := \{U_1, U_2, \dots U_k\}$ minimizing $\sum_{p \in P} \min_{U_i \in \mathcal{U}} \|(I - U_i U_i^T)p\|_2^z$.

In this chapter our goal is to address the following questions:

RQ3 *How does the excess risk behave as function of k, d, and n for center-based clustering ?*

RQ4 *And how does the excess risk behave as function of k, d, j, and n for subspace clustering ?*

Special cases include $(k, 1)$ clustering, known as k-median, $(k, 2)$ clustering known as k-means and $(k, j, 2)$ clustering known as projective clustering. Generally, there seems to be an interest in varying z, as letting z tend towards 1 tends to result in outlier-robust clusterings. The problem is less widely explored for $z > 2$, although it still has numerous applications. For example, centralised moments with respect to the three and four norms are skewness and kurtosis, respectively, and are extensively employed in statistics. Fitting a mixture model with respect to skewness minimizes asymmetry around the target center. Higher powers enable us to put a greater emphasis on outliers in a more tractable manner, see for example Cohen-Addad et al. (2021), as $z \to \infty$ tends to suffer heavily from the curse of dimensionality (Agarwal et al., 2004; Ding, 2020). Despite a huge interest and a substantial amount of research, so far optimal risk bounds $\tilde{O}\left(\sqrt{k/n}\right)$[1] for the k-means problem have been established, see the seminal paper by Fefferman et al. (2016) for the upper bound and Bartlett et al. (1998c) for nearly matching lower bounds. For general (k, z)-clustering problems, the best known results prove a risk bound of $O\left(\sqrt{kd/n}\right)$

[1] $\tilde{O}$ hides logarithmic terms, i.e. we consider $O\left(\sqrt{k/n} \cdot \text{polylog}(k, n)\right) = \tilde{O}\left(\sqrt{k/n}\right)$.

(Bartlett et al., 1998c). For $(k, j, 2)$ clustering, the best known bounds of $\tilde{O}\left(\sqrt{kj/n}\right)$ are due to Fefferman et al. (2016). Thus, the following question naturally arises:

Is it possible to obtain optimal generalization bounds for all (k, j, z)-clustering objectives?

We answer this question in the affirmative whenever j and z are constant. Specifically, we show

- The excess risk bound for (k, z)-clustering when given n independent samples from an unknown fixed distribution $\mathcal{D}$ is bounded by $\tilde{O}\left(\sqrt{k/n}\right)$, matching the lower bound of Bartlett et al. (1998c).
- The excess risk bound for (k, j, z)-clustering when given n independent samples from an unknown fixed distribution $\mathcal{D}$ is bounded by $\tilde{O}\left(\sqrt{kj^2/n}\right)$.
- There exists a distribution such that the excess risk for the $(k, j, 2)$-clustering problem is at least $\Omega\left(\sqrt{kj/n}\right)$, matching the upper bound of Fefferman et al. (2016) up to polylog factors.

3.0.1 Related work

The most basic question one could answer is if the empirical estimation performed on P is consistent, i.e. as $n \to \infty$, whether the excess risk tends to 0. This was shown in a series of works by Pollard (Pollard, 1981, 1982b), see also Abaya and Wise (1984). Subsequent work then analyzed the convergence rate of the risk. The first works in this direction proved convergence rates of the order $\tilde{O}(1/\sqrt{n})$ without giving dependencies on other parameters (Chou, 1994; Pollard, 1982a). Linder et al. (1994) gave an upper bound of $O(d^{3/2}\sqrt{k/n})$. Linder (2000) improved the upper bound to $O(d\sqrt{k/n})$. Bartlett et al. (1998a) showed an upper bound $O(\sqrt{kd/n})$ and gave a lower bound of $\Omega(\sqrt{k^{1-4/d}/n})$. Motivated by applications of clustering for high dimensional kernel spaces (Bach and Jordan, 2005; Calandriello and Rosasco, 2018; Chitta et al., 2011, 2012; Dhillon et al., 2004; Fefferman et al., 2016; Liu et al., 2020, 2019; Oglic and Gärtner, 2017; Rudi and Rosasco, 2016; Wang et al., 2019a; Yin et al., 2020; Zhang and Liao, 2019), research subsequently turned its efforts towards minimizing the dependency on the dimension. Biau et al. (2008b) presented an upper bound of $O(k/\sqrt{n})$, see also the work by Clémençcon (Clémençcon, 2011). Fefferman et al. (2016) gave a matching upper bound of the order $O(\sqrt{k/n})$, which was later recovered by using techniques from Foster and Rakhlin Foster and Rakhlin (2019) and Liu (Liu, 2021). Further improvements require additional assumptions of the distribution $\mathcal{D}$, see Antos et al. (2005); Levrard (2015). For subspace clustering, there have only been results published for the case $z = 2$ (Fefferman et al., 2016; Lauer, 2020; Shawe-Taylor et al., 2005), for which the state of the art provides a $\tilde{O}\left(\sqrt{kj/n}\right)$ risk bound due to Fefferman et al. (2016). A highly related line of research originated with the study of coresets for compression. For Euclidean (k, z) clustering, coresets with space bounds of $\tilde{O}\left(k/\varepsilon^{2+z}\right)$ have been established (Cohen-Addad et al., 2021b, 2022a), which roughly corresponds to a error rate of $\tilde{O}\left(\sqrt[2+z]{k/n}\right)$ as a function of the size of the compression. For the specific case of k-median and k-means, coresets with space bounds of $\tilde{O}\left(k^{(2z+2)/(z+2)}/\varepsilon^2\right)$ are known (Cohen-Addad et al., 2022b), which corresponds to a error rate of $\tilde{O}\left(\sqrt{k^{(2z+2)/(z+2)}/n}\right)$. Both results are optimal for certain ranges of ε and k (Huang et al., 2022) and while these bounds are worse than what we hope to achieve for generalization, many of the techniques such as terminal embeddings are relevant for both fields. For (k, j, z) clustering, coresets are only known to exist under certain assumptions, where the provable size is $\tilde{O}\left(\exp(k, j, \varepsilon^{-1})\right)$ (Feldman and Langberg, 2011; Feldman et al., 2020).

3.0.2 Notation

We use $\|x\|_p := \sqrt[p]{\sum |x_i|^p}$ to denote the ℓ_p norm of a vector x. For $p \to \infty$, we define the limiting norm $\|x\|_\infty = \max x_i$. Further, we refer to the d-dimensional unit Euclidean ball by B_2^d, i.e. $x \in B_2^d$ is a vector in $\mathbb{R}^d$ and $\|x\|_2 := \sqrt{\sum_{i=1}^d x_i^2} \leq 1$. Let U be a $d \times j$ orthogonal matrix, i.e., with columns that are pairwise orthogonal and have unit Euclidean norm. We say that UU^T is the projection matrix associated with U. Let z be a positive integer. Given any set $\mathcal{S}$ of k points in B_2^d we denote the (k, z)-clustering cost for a point set P with respect to solution $\mathcal{S}$ as

$$\text{cost}(P, \mathcal{S}) := \sum_{p \in P} \min_{s \in \mathcal{S}} \|p - s\|_2^z.$$

Special cases include k-means ($z = 2$) and k-median ($z = 1$). Similarly, given a collection $\mathcal{U}$ of k orthogonal matrices of rank at most j, we denote the (k, j, z)-clustering cost of a point set P as

$$\text{cost}(P, \mathcal{U}) := \sum_{p \in P} \min_{U \in \mathcal{U}} \|(I - UU^T)p\|_2^z.$$

The specific case $(k, j, 2)$ is often known as projective clustering in literature. The cost vector $v^{\mathcal{S},P} \in \mathbb{R}^{|P|}$, respectively $v^{\mathcal{U},P} \in \mathbb{R}^{|P|}$ has entries $v_p^{\mathcal{S}} = \min_{s \in \mathcal{S}} \|p - s\|_2^z$, respectively $v_p^{\mathcal{U}} = \min_{U \in \mathcal{U}} \|(I - UU^T)p\|_2^z$ for $p \in P$. We will omit P from $v^{\mathcal{S},P}$ and $v^{\mathcal{U},P}$, if P is clear from context. The overall cost is $\|v^{\mathcal{S}}\|_1 = \sum_{p \in P} \min_{s \in \mathcal{S}} \|p - s\|_2^z$ and $\|v^{\mathcal{U}}\|_1 = \sum_{p \in P} \min_{U \in \mathcal{U}} \|(I - UU^T)p\|_2^z$. The set of all cost vectors is denoted by V.

Let $\mathcal{D}$ be an unknown but fixed distribution on B_2^d with probability density function $\mathbb{P}$. For any solution $\mathcal{S}$, respectively $\mathcal{U}$, we define $\text{cost}(\mathcal{D}, \mathcal{S}) := \int_{p \in B_2^d} \min_{s \in \mathcal{S}} \|p - s\|^z \cdot \mathbb{P}[p] dp$ and $OPT := \min_{\mathcal{S}} \text{cost}(\mathcal{D}, \mathcal{S})$ and respectively $\text{cost}(\mathcal{D}, \mathcal{U}) := \int_{p \in B_2^d} \min_{U \in \mathcal{U}} \|(I - UU^T)p\|^z \cdot \mathbb{P}[p] dp$ and $OPT := \min_{\mathcal{U}} \text{cost}(\mathcal{D}, \mathcal{U})$. Let P be a set of n points sampled independently from $\mathcal{D}$. We denote the cost of the empirical risk minimizer on P by $OPT_P := \frac{1}{n} \min_{\mathcal{S}} \|v^{\mathcal{S}}\|_1$, and respectively, $OPT_P := \frac{1}{n} \min_{\mathcal{U}} \|v^{\mathcal{U}}\|_1$. The excess risk of P with respect to a set of cost vectors is denoted by

$$\mathcal{E}_{|P|}(V) := \mathbb{E}_P[OPT_P] - OPT.$$

Finally, we use the notion of a net. Let $(V, dist)$ be a metric space, $\mathcal{N}(V, dist, \varepsilon)$ is an ε-net of the set of vectors V, if for all $v \in V$ $\exists$ $v' \in \mathcal{N}(V, dist, \varepsilon)$ such that $dist(v, v') \leq \varepsilon$. We will particularly focus on nets for cost vectors induced by (k, z)-clustering and (k, j, z)-clustering defined as follows, prior work has proposed similar nets for coresets and sublinear algorithms for (k, z) clustering (Cohen-Addad et al., 2021a).

Definition 3.0.1 (Clustering Nets)**.** A set $\mathcal{N}_\varepsilon$ of $|P|$-dimensional vectors is an ε-clustering net if for every cost vector v obtained from a solution $\mathcal{S}$ or $\mathcal{U}$, there exists a vector $v' \in \mathcal{N}_\varepsilon$ with $\|v' - v\|_\infty \leq \varepsilon$

3.1 Outline and technical contribution

for each section, we decided to first state the theorem and present an overview of the theory and then provide the the full proofs in the following subsection. In this section, we endeavour to present a complete and accessible overview of the key ideas behind the theorems. Let P be a set of n points sampled independently from some unknown but fixed distribution $\mathcal{D}$. To show that the excessive risk with respect to clustering objectives is in $\tilde{O}(f(n))$ for some function f, it is sufficient to show two things. First, that for the optimal solution $\mathcal{U}_{\text{OPT}}$, the clustering cost estimated using P is close to the true cost. Second, any solution that is more expensive than $\mathcal{U}_{\text{OPT}}$ does not become too cheap when evaluated on P. Both conditions are satisfied if for any solution $\mathcal{U}$

$$\left|\frac{1}{n}\text{cost}(P,\mathcal{U}) - \text{cost}(\mathcal{D},\mathcal{U})\right| \in O(f(n)).$$

Showing $\mathbb{E}_P \left|\frac{1}{n}\text{cost}(P,\mathcal{U}_{\text{OPT}}) - \text{cost}(\mathcal{D},\mathcal{U}_{\text{OPT}})\right| \in O(\sqrt{1/n})$ is typically a straightforward application of concentration bounds such as Chernoff's bound. In fact, these concentration bounds show something even stronger. Given t solutions $\mathcal{U}_1, \ldots \mathcal{U}_t$, we have

$$\mathbb{E}_P \sup_{\mathcal{U}_i} \left|\frac{1}{n}\text{cost}(P,\mathcal{U}_i) - \text{cost}(\mathcal{D},\mathcal{U}_i)\right| \in O\left(\sqrt{\log t/n}\right). \tag{3.1}$$

What remains is to bound the number of solutions t.

Clustering nets and dimension reduction for center based clustering Unfortunately, the total number of expensive clusterings in Euclidean space is infinite, making a straightforward application of 3.1 useless. Nets as per Definition 3.0.1 are now typically used to reduce the infinite number of solutions to a finite number. Specifically, one has to show that by preserving the costs of all solutions in the net, the cost of any other solution is also preserved. Using basic techniques from high dimensional computational geometry, it is readily possible to prove that a ε-net for (k, j, z) clustering of size $\exp(k \cdot j \cdot d \cdot \log \varepsilon^{-1})$ exists, where d is the dimension of the ambient space. Plugging this into Equation 3.1 and setting $\varepsilon^{-1} = n$ then yields a generalization bound of the order $O\left(\sqrt{kjd \log n/n}\right)$. Unfortunately, this leads to a dependency on d, which is suboptimal. To improve the upper bounds, we take inspiration from coreset research. For (k, z)-clustering, a number of works have investigated dimension reduction techniques known as terminal embeddings, see Becchetti et al. (2019); Huang et al. (2021). Given a set of points $P \in \mathbb{R}^d$, a terminal embedding $f : \mathbb{R}^d \to \mathbb{R}^m$ guarantees $\|p - q\|_2 = (1 \pm \varepsilon) \cdot \|f(p) - f(q)\|_2$ for any $p \in P$ and $q \in \mathbb{R}^d$. Terminal embeddings are very closely related to the Johnson-Lindenstrauss lemma, but more powerful in key regard: only one of the points is required to be in P. The added guarantee extended to arbitrary $q \in \mathbb{R}^d$ due to terminal embeddings allows us to capture all possible solutions. There are also even simpler proofs for k-mean that avoid this machinery entirely, see Fefferman et al. (2016); Foster and Rakhlin (2019); Liu (2021). Unfortunately, these arguments are heavily reliant on properties of inner products and are difficult to extend to other values of z. The terminal embedding technique may be readily adapted to

(k, z)-clustering, though some care in the analysis must be made to avoid the worse dependencies on the sample size necessitated for the corset guarantee, described as follows.

Improving the union bound via chaining: To illustrate the chaining technique, consider the simple application of the union bound for a terminal embedding with target dimension $m = \Theta(\varepsilon^{-2} \log n)$, see the main result of Narayanan and Nelson (2019). Replacing the dependency on d with an appropriately chosen parameters and plugging the resulting net N_ε of size $\exp(k\varepsilon^{-2})$ yields a generalization bound of $O\left(\sqrt[4]{k \log n/n}\right)$ for (k, z) clustering. We improve on this using a chaining analysis, see Cohen-Addad et al. (2021b, 2022a) for its application to coresets for (k, z) clustering and Fefferman et al. (2016) for $(k, j, 2)$ clusterings. Specifically, we use a nested sequence of nets $N_{1/2}, N_{1/4}, N_{1/8}, \ldots, N_{2^{-2\log n}}$. Note that for every solution $\mathcal{S}$, we may now write $\text{cost}(p, \mathcal{S})$ for any $p \in P$ as a telescoping sum

$$\text{cost}(p, \mathcal{S}) = \sum_{h=0}^{\infty} \text{cost}(p, \mathcal{S}_{2^{-(h+1)}}) - \text{cost}(p, \mathcal{S}_{2^{-h}})$$

with $,\mathcal{S}_{2^{-h}} \in N_h$ and $\text{cost}(p, \mathcal{S}_1)$ being set to 0. We use this as follows. Suppose for some solution $\mathcal{S}$, we have solutions $\mathcal{S}_{2^{-h}} \in N_{2^{-h}}$ and $\mathcal{S}_{2^{-(h+1)}} \in N_{2^{-(h+1)}}$. Then $|\text{cost}(p, \mathcal{S}_{2^{-h}}) - \text{cost}(p, \mathcal{S}_{2^{-(h+1)}})| \leq O(2^{-h}) \, |\text{cost}(p, \mathcal{S}_{2^{-h}}) - \text{cost}(p, \mathcal{S})|$ for all $p \in P$. Instead of applying the union bound for a small set of solutions, we apply the union bound along every pair of solutions appearing in the telescoping sum. Using arguments similar to Equation 3.1, we then obtain

$$\mathbb{E}_P \sup_{\substack{\mathcal{S}_{2^{-h}} \times \mathcal{S}_{2^{-(h+1)}} \\ \in N_h \times N_{h+1}}} \left| \frac{1}{n}\text{cost}(P, \mathcal{S}_{2^{-h}}) - \frac{1}{n}\text{cost}(P, \mathcal{S}_{2^{-(h+1)}}) \right|$$

$$= 2^{-h} \cdot \tilde{O}\left(\sqrt{\frac{\log(|N_h| \cdot |N_{h+1}|)}{n}}\right) = 2^{-h} \cdot \tilde{O}\left(\sqrt{\frac{k \cdot 2^{2h} \cdot \text{polylog}(k/2^h)}{n}}\right) \in \tilde{O}\left(\sqrt{\frac{k}{n}}\right)$$

This is the desired risk bound for (k, z) clustering. To complete the argument in a rigorous fashion, we must now merely combine the decomposition of $\text{cost}(P, \mathcal{S})$ into the telescoping sum with the learning rate that we just derived. Indeed, this already provides a simple way of obtaining a bound on the risk of the order $\tilde{O}\left(\sqrt{k/n}\right)$, which turns out to be optimal. In summary, to apply the chaining technique successfully, we require two properties: (i) the dependency on ε in the net size can be at most $\exp(\tilde{O}(\varepsilon^{-2}))$, as the increase in net size is then met with a corresponding decrease between successive estimates along the chain and (ii) the nets have to preserve the cost up to an additive ε for *every* sample point p. The second property is captured by Definition 3.0.1. Both properties impose restrictions on the dimension reductions that can be successfully integrated into the chaining analysis.

Dimension reduction for projective clustering: It turns out that extending this analysis (k, j, z) clustering is a major obstacle. While the chaining method itself uses no particular properties of (k, z) clustering, the terminal embeddings needed to obtain nets cannot be applied to subspaces. Indeed, terminal embeddings by the very nature of their guarantee, cannot be linear[2], and hence a linear structure such

[2]Consider an embedding matrix $S \in \mathbb{R}^{d \times m}$. Clearly, there exists some vector $x \in \mathbb{R}^d$ that is in the kernel of S whenever $m < d$, hence for any vector p, $\|p - (x + p)\|_2$ cannot be preserved.

as a subspace will not be preserved. At this stage, there are a number of initially promising candidates that can provide alternative dimension reduction methods. For example, the classic Johnson-Lindenstrauss lemma can be realized via a random embedding matrix and, moreover, preserves subspaces, see for example Sarlós (2006); Clarkson and Woodruff (2009); Cohen et al. (2015). Unfortunately, as remarked by Huang et al. (2021), there is an inherent difficulty in applying Johnson-Lindenstrauss type embeddings even for (k, z) clustering coresets and the same arguments also apply for generalization bounds.

An alternative dimension reduction method based on principal component analysis was initially proposed by Feldman et al. (2020) for $(k, j, 2)$, see also Cohen et al. (2015) and most notably Sohler and Woodruff (2018) for a different variant that applies to arbitrary (k, j, z) objectives. For $(k, j, 2)$ clustering, it states that a dimension reduction on the first $O(D/\varepsilon)$ principal components preserves the projective cost of all subspaces of dimension D. Since $(k, j, 2)$ clustering is a special case of a $k \cdot j$ dimensional projection, it implies that $O(kj/\varepsilon)$ dimensions are sufficient. Given that these dimension reductions are based on PCA-type methods, they are linear and therefore seem promising initially. Unfortunately, this technique has serious drawbacks. It does not satisfy the requirements for Definition 3.0.1, only preserving the cost on aggregate rather then per individual point, and thus cannot be combined with the chaining technique. Without the chaining technique, the best bound one can hope for is of the order $\tilde{O}\left(\sqrt[3]{k^2 j^2/n}\right)$, which falls short of what we are aiming for.

Another important technique used to quantify optimal solutions of (k, j, z) clustering initially proposed by Shyamalkumar and Varadarajan (2007) and subsequently explored by Feldman et al. (2010); Deshpande and Varadarajan (2007) and has frequently seen use in coreset literature (Feldman and Langberg, 2011; Huang et al., 2021). Succinctly, it states that a $(1 + \varepsilon)$ approximate solution to the $(1, j, z)$ clustering problem of a point set P is contained in a subspace spanned by $\tilde{O}(j^2/\varepsilon)$ input points of P. While this result improves over PCA for large values of k, applying it only yields a learning rate of the order $O(\sqrt[3]{kj^3/n})$. It turns out that this technique has the exact same limitations as PCA, namely that costs per point are not preserved, and thus only offers a different tradeoff in parameters.

Our new insight: Given the state of the art, designing a dimension reduction technique that would enable the application of the chaining technique might seem hopeless, and indeed, we were not able to prove such. The key insight that allows us to bypass these bottlenecks is to find a dimension reduction that applies not to all solutions $\mathcal{U}$, but only to a certain subset of them. Indeed, we show that for any point set P contained in the unit ball and any subspace $\mathcal{U}$ of dimension j, there exists a subspace S spanned by $O(j/\varepsilon^2)$ points of P such that for every point p: $|\text{cost}(p, \mathcal{U}) - \text{cost}(p_S, \mathcal{U}_S)| \le \varepsilon$. This is similar to the guarantee provided by Shyamalkumar and Varadarajan (2007) but stronger in that it (i) applies to arbitrary subspaces, which is required for the chaining analysis, and (ii) applies to each point of P individually, rather than for the entire point set P on aggregate. We then augment the chaining analysis by applying a union bound over all $\binom{|P|}{j/\varepsilon^2}$ possible dimension reductions, thereby capturing all solutions $\mathcal{U}$. We are unaware of any previously successful attempts at integrating multiple dimension reductions within a chaining analysis and believe that the technique may be of independent interest.

3.2 Useful results from learning theory

Our goal is to bound the rate with which the empirical risk decreases for clustering problems. For a fixed set of n points P and a set of functions $F : P \to \mathbb{R}$, we define the Rademacher complexity (Rad_n) and the Gaussian complexity (G_n) wrt F respectively as

$$Rad_n(F) = \frac{1}{n} \cdot \mathbb{E}_r \sup_{f \in F} \sum_{p \in P} f(p) \cdot r_p \qquad G_n(F) = \frac{1}{n} \cdot \mathbb{E}_g \sup_{f \in F} \sum_{p \in P} f(p) \cdot g_p$$

where r_p are independent random variables following the Rademacher distribution, whereas g_p are independent Gaussian random variables. In our case, we can think of f as being associated to a solution $\mathcal{S}$ (respectively a solution $\mathcal{U}$) and $f(p) = \text{cost}(p, \mathcal{S}) = \min_{s \in \mathcal{S}} \|p - s\|_2^z$ (respectively $f(p) = \text{cost}(p, \mathcal{U}) = \min_{U \in \mathcal{U}} \|(I - UU^T)p\|_2^z$). Since we associate every f with a cost vector $v^{\mathcal{S}}$, we will use $Rad_n(F)$ and $Rad_n(V)$ as well as $G_n(F)$ and $G_n(V)$ interchangeably. The following theorem is due to Bartlett and Mendelson (Bartlett and Mendelson, 2002).

Theorem 3.2.1 (Simplified variant of Theorem 8 of Bartlett and Mendelson (2002))**.** *Consider a loss function $L : A \to [0, 1]$. Let F be a class of functions mapping from X to A and let $(X_i)_{i=1}^n$ be independent samples from $\mathcal{D}$. Then, for any integer n and any $\delta > 0$, with probability at least $1 - \delta$ over samples of length n, denoting by $\hat{\mathbb{E}}_n$ the empirical risk, every $f \in F$ satisfies*

$$\mathbb{E}L(f(X)) \leq \hat{\mathbb{E}}_n L(f(X)) + Rad_n(F) + \sqrt{\frac{8 \ln 2/\delta}{n}}.$$

Thus, in order to bound the excess risk, Theorem 3.2.1 shows that it is sufficient to bound the Rademacher complexity. It is well known (see, for example, B.3 of Rudra and Wootters (2014)) that $Rad_n(V) \leq \sqrt{2\pi} G_n(V)$. Thus we can alternatively bound the Gaussian complexity, which is sometimes more convenient. Note that if V is the set of all cost vectors, clustering nets are mere $\mathcal{N}(V, \|.\|_\infty, \varepsilon)$. Using these nets, we can bound the Rademacher and Gaussian complexity. Indeed the following lemma holds.

Lemma 3.2.2. *Let $\mathcal{D}$ be a distribution over B_2^d and let P a set of n points sampled from $\mathcal{D}$. Suppose that for a set of n-dimensional vector V, we have an absolute constants $C, \gamma > 0$ such that*

$$\log |\mathcal{N}(V, \|.\|_\infty, \varepsilon)| \in O(\varepsilon^{-2} \log^\gamma(n\varepsilon^{-1}) C). \tag{3.2}$$

Then

$$G_n(V) \in O\left(\sqrt{\frac{C \log^{\gamma+2} n}{n}}\right).$$

Proof of Lemma 3.2.2

In this section, we include the proof of Lemma 3.2.2 and some preliminary facts that will be useful for the proof.

Let r be a Rademacher vector, i.e. every entry r_i is sampled independently uniformly from $\{-1, 1\}$. Further, we say that g is a Gaussian vector if every entry g_i is a standard Gaussian with mean 0 and variance 1. We have the following useful properties of Gaussians.

Fact 3.2.2.1 (Appendix B.1 by Rudra and Wootters (2014)). *Let $g_1, \dots g_n$ be Gaussians with means μ_i and variances σ_i^2.*

- *If $\sigma_i^2 \leq \sigma^2$ for all i, then $\mathbb{E}[\max_{g_i} |g_i|] \leq 2\sigma\sqrt{2\log n}$.*
- *If the Gaussians are independent, then $\sum_{i=1}^n a_i g_i$ is Gaussian distributed with mean $\sum_{i=1}^n a_i\mu_i$ and variance $\sum_{i=1}^n a_i^2\sigma_i^2$.*
- *If the g_i are independent standard Gaussians with mean 0 and variance 1, then $Y := \sum_{i=1}^n g_i^2$ is Chi-squared distributed with mean $\mathbb{E}[\sqrt{Y}] \in O(\sqrt{n})$.*

Another result we need is the following

Lemma 3.2.3 (Lemma 5.2 of Vershynin (2012)). $|\mathcal{N}(B_2^d, \|.\|_2, \varepsilon)| \leq (1 + 2/\varepsilon)^d$.

We are now ready to prove the Lemma 3.2.2. The proof of Lemma is similar to arguments used to prove Dudley's theorem.

Proof Lemma 3.2.2. For ease of notation, we use solutions $\mathcal{S}$ induced by points, but the proof carries over without any modifications other than changing the notation to collections of subspaces $\mathcal{U}$.

Consider an arbitrary cost vector $v^{\mathcal{S}}$. We write $v^{\mathcal{S}}$ as a telescoping sum

$$v^{\mathcal{S}} := \sum_{h=0}^{\infty} v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}}$$

where $v^0 = 0$ and $v^{i,\mathcal{S}}$ is a vector from $\mathcal{N}(V, \|.\|_\infty, 2^{-i})$ approximating $v^{\mathcal{S}}$. Observe that

$$\|v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}}\|_\infty \leq \|v^{h+1,\mathcal{S}} - v^{\mathcal{S}} + v^{\mathcal{S}} - v^{h,\mathcal{S}}\|_\infty \leq 2 \cdot 2^{-h} \tag{3.3}$$

due to the triangle inequality. Thus we have

$$\begin{aligned}
n \cdot G_n(V) &= \mathbb{E}_{P,g}\left[\sup_{\mathcal{S}} (v^{\mathcal{S}})^T g\right] = \mathbb{E}_{P,g}\left[\sup_{\mathcal{S}} \sum_{h=0}^{\infty} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g\right] \\
&\leq \mathbb{E}_{P,g} \sum_{h=0}^{\infty} \left[\sup_{\mathcal{S}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g\right] \\
&= \mathbb{E}_{P,g} \sum_{h=0}^{\infty} \left[\sup_{\substack{v^{h+1,\mathcal{S}}, v^{h,\mathcal{S}} \in \\ \mathcal{N}(V,\|\cdot\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V,\|\cdot\|_\infty,2^{-h})}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g\right] \\
&= \mathbb{E}_{P,g} \sum_{h=0}^{\log n} \left[\sup_{\substack{v^{h+1,\mathcal{S}}, v^{h,\mathcal{S}} \in \\ \mathcal{N}(V,\|\cdot\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V,\|\cdot\|_\infty,2^{-h})}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g\right] \\
&\quad + \mathbb{E}_{P,g} \sum_{h=\log n}^{\infty} \left[\sup_{\substack{v^{h+1,\mathcal{S}}, v^{h,\mathcal{S}} \in \\ \mathcal{N}(V,\|\cdot\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V,\|\cdot\|_\infty,2^{-h})}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g\right]
\end{aligned}$$

$$= \mathbb{E}_{P,g} \sum_{h=0}^{\log n} \left[\sup_{\substack{v^{h+1,\mathcal{S}}, v^{h,\mathcal{S}} \in \\ \mathcal{N}(V,\|.\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V,\|.\|_\infty, 2^{-h})}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g \right] \tag{3.4}$$

$$+\mathbb{E}_{P,g} \left[\sup_{\mathcal{S}} (v^{\mathcal{S}} - v^{\log n,\mathcal{S}})^T g \right] \tag{3.5}$$

We bound the terms 3.4 and 3.5 differently, starting with the latter.

For every $\mathcal{S}$

$$(v^{\mathcal{S}} - v^{\log n,\mathcal{S}})^T g \leq \|v^{\mathcal{S}} - v^{\log n,\mathcal{S}}\|_2 \cdot \mathbb{E}[\|g\|_2],$$

due to the Cauchy Schwarz inequality. Further,

$$\|v^{\mathcal{S}} - v^{\log n,\mathcal{S}}\|_2 \leq \sqrt{n} \cdot \|v^{\mathcal{S}} - v^{\log n,\mathcal{S}}\|_\infty \leq \sqrt{n} \cdot 2^{-\log n} = \sqrt{\frac{1}{n}},$$

which, combined with the third item in Fact 3.2.2.1 yields

$$\mathbb{E}_{P,g} \left[\sup_{\mathcal{S}} (v^{\mathcal{S}} - v^{\log n,\mathcal{S}})^T g \right] \in O\left(\sqrt{\frac{1}{n}} \cdot \sqrt{n} \right) = O(1). \tag{3.6}$$

We now consider the term 3.4. Due to the second item of Fact 3.2.2.1, $(v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g$ is Gaussian distributed with mean 0 and variance

$$\sum_{i=1}^{n} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})_i^2 \leq 4n \cdot 2^{-2h}.$$

Thus, we have, using the first item in Fact 3.2.2.1

$$\mathbb{E}_{P,g} \sum_{h=0}^{\log n} \left[\sup_{\substack{v^{h+1,\mathcal{S}}, v^{h,\mathcal{S}} \in \\ \mathcal{N}(V,\|.\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V,\|.\|_\infty, 2^{-h})}} (v^{h+1,\mathcal{S}} - v^{h,\mathcal{S}})^T g \right]$$
$$\leq \sum_{h=0}^{\log n} \sqrt{32n \cdot 2^{-2h} \log |\mathcal{N}(V, \|.\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V, \|.\|_\infty, 2^{-h})|}$$

Now using equation (3.2) we obtain that,

$$32n \cdot 2^{-2h} \log \left| \mathcal{N}(V, \|.\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V, \|\dot{|}_\infty, 2^{-h}) \right| \in O(n \cdot \log^\gamma n)$$

So we have that

$$\sum_{h=0}^{\log n} \sqrt{32n \cdot 2^{-2h} \log |\mathcal{N}(V, \|.\|_\infty, 2^{-(h+1)}) \times \mathcal{N}(V, \|.\|_\infty, 2^{-h})|} \in O(\sqrt{n \cdot \log^{\gamma+2} n}) \tag{3.7}$$

Adding the bounds (3.7) and (3.6) for Terms (3.5) and (3.4), respectively yields the claim. □

Finally, we will frequently use the following triangle inequality extended to powers.

Lemma 3.2.4 (Triangle Inequality for Powers (Lemma A.1 of Makarychev et al. (2019)))**.** *Let a, b, c be an arbitrary set of points in a metric space with distance function d and let z be a positive integer. Then for any $\varepsilon > 0$*

$$d(a,b)^z \leq (1+\varepsilon)^{z-1} d(a,c)^z + \left(\frac{1+\varepsilon}{\varepsilon}\right)^{z-1} d(b,c)^z$$

$$|d(a,b)^z - d(a,c)^z| \leq \varepsilon \cdot d(a,c)^z + \left(\frac{2z+\varepsilon}{\varepsilon}\right)^{z-1} d(b,c)^z.$$

The specific types of nets used in our study and the size bounds for those nets will be the key to obtaining the desired upper bounds and will be detailed in the next section.

3.3 Generalization bounds for center-based clustering and subspace clustering

We start by giving our generalization bounds for center based clustering and subspace clustering problems. For subspace clustering problems, we first state the result for general (k, j, z) clustering. An improvement for the special case $z = 2$ will be given later.

Theorem 3.3.1. *Let $\mathcal{D}$ be a distribution over B_2^d and let P be a set of n points sampled from $\mathcal{D}$. For any set of k points $\mathcal{S} \subset B_2^d$, we denote by $v^{\mathcal{S}}$ the n-dimensional cost vector of P in solution $\mathcal{S}$ with respect to the (k, z)-clustering objective. Moreover we denote by $v^{\mathcal{U}}$ the n-dimensional cost vector of P in solution $\mathcal{U}$ with respect to the (k, j, z)-clustering objective. Let V_z be the union of all cost vectors of P for the center-based clustering and $V_{j,z}$ the union of all cost vectors for subspace clustering. Then with probability at least $1 - \delta$*

$$\mathcal{E}_n(V_z) \in O\left(\sqrt{\frac{k \cdot \log^4 n}{n}} + \sqrt{\frac{\log 1/\delta}{n}}\right) \tag{3.8}$$

$$\mathcal{E}_n(V_{j,z}) \in O\left(\sqrt{\frac{k \cdot j^2 \cdot \log jn \cdot log^3 n}{n}} + \sqrt{\frac{\log 1/\delta}{n}}\right). \tag{3.9}$$

Following Theorem 3.2.1, it is sufficient to bound the Rademacher complexity in order to bound the excess risk. The Rademacher complexity is, up to lower order terms, equal to the Gaussian complexity, which, following Lemma 3.2.2 may be bounded by obtaining small nets with respect to the $\|.\|_\infty$ norm. We believe that the results on the bounds of the nets, may be of independent interest and we'll state these results in the following Lemma.

Lemma 3.3.2. *Let $\mathcal{D}$ be a distribution over B_2^d and let P a set of n points sampled from $\mathcal{D}$, let V_z be defined as in Theorem 3.3.1 let $V_{j,z}$ be defined as in Theorem 3.3.1. Then*

$$|\mathcal{N}(V_z, \|.\|_\infty, \varepsilon)| \leq \exp(O(1) z^3 \cdot k \cdot \varepsilon^{-2} \log n \cdot (\log(z) + \log(\varepsilon^{-1}))) \tag{3.10}$$

$$|\mathcal{N}(V_{j,z}, \|.\|_\infty, \varepsilon)| \leq \exp(O(1)(3z)^{z+2} \cdot k \cdot j \cdot \varepsilon^{-2}(\log n + j \log(j\varepsilon^{-1})) \log \varepsilon^{-1}). \tag{3.11}$$

Combining Lemma 3.3.2 with Lemma 3.2.2 now yields the immediate bound on the Rademacher and Gaussian complexity. Following the discussion from Section 3.1, we use terminal embeddings to prove the part of Lemma 3.3.2 pertaining to (k, z) clustering, see Sec. 3.3.1. Unfortunately, the terminal embedding technique is not admissible for obtaining nets for subspace clustering as clarified in Section 3.1. Thus, we use an entirely different approach. We show the existence of a collection of dimension reducing maps with subspace preserving properties. Fortunately, the number of dimension reducing maps is small. Our desired net sizes then follow by enumerating over all of these dimension reducing maps, and for the candidate solutions covered by each such dimension reducing map, we can find an efficient net. First, we introduce a slightly different, but closely related notion to $(1, j, z)$-nets.

Definition 3.3.3 (Projective Nets)**.** Let $P \subset B_2^d$ be a set of points, and let z be a positive integer. For a $d \times j$ matrix S with columns that have at most unit norm and any point $p \in P$, define the projective cost as $\text{cost}_{proj}(p, S) = \|S^T p\|_2$. Let V be the set of all projective cost vectors induced by such matrix S. We call a $\mathcal{N}(V, \|.\|_\infty, \varepsilon)$ a (ε, j)-projective net of P.

On a high level, the proof largely relies on the following decomposition. Let U be a candidate subspace and let Π be a projection matrix used to approximate $\|(I - UU^T)p\|_2^z$ We have

$$\begin{aligned}\|(I - UU^T)p\|^2 = &\underbrace{\|\Pi p\|^2}_{(1)} - \underbrace{\|U^T \Pi p\|^2}_{(2)} + \underbrace{\|(I - \Pi)p\|^2}_{(3)} \\ &- \underbrace{\|UU^T(I - \Pi)p\|^2}_{(4)} + \underbrace{2p^T \Pi UU^T(I - \Pi)p}_{(5)} \end{aligned} \tag{3.12}$$

Here, we wish to select Π such that $\|U^T(I - \Pi)p\|_2$ is small for all $p \in P$. Note that this implies that the terms $2p^T \Pi UU^T(I - \Pi)p$ and $\|UU^T(I - \Pi)p\|^2$ are small. For the term (2), we merely have to show that projective nets exist. If the number of Π is small, we can further construct good nets for the terms (1) and (3) . We start by giving a bound for the projective nets. Our first Lemma 3.3.4 shows that if the points lie in a sufficiently low-dimensional space, such a net can be obtained by constructing a net $\mathcal{N}(B_2^d, \|.\|_2, \varepsilon')$ for a sufficiently small ε'.

Lemma 3.3.4. *Let $P \subset B_2^d$ be a set of points and let z be a positive integer. Then there exists an (ε, j)-projective net of size $|\mathcal{N}(V, \|.\|_\infty, \varepsilon)| \leq \exp(O(1) \cdot d \cdot j \cdot \log(j\varepsilon^{-1}))$.*

To reduce the dependency on the dimension, we now use the following lemma. Essentially, it shows that in order to retain the properties of U, we can find a projection matrix Π of rank at most $O(j\varepsilon^{-2})$.

Lemma 3.3.5. *Let $P \subseteq B_2^d$. For any orthogonal matrix $U \in \mathbb{R}^{j \times d}$, there exists $M \subseteq P$, with $|M| \in O(j \cdot \varepsilon^{-2})$, such that $\forall p \in P, \|U^T(I - \Pi_M)p\| \leq \varepsilon \cdot \|(I - \Pi_M)p\|$.*

We now use this lemma as follows. We can efficiently enumerate over all candidate Π. This immediately gives us 0-nets for the terms (1) and (3). For each Π, we then apply Lemma 3.3.4, which gives us a net for term (2). Finally, by choice of Π, we can show that terms (4) and (5) are negligible.

3.3.1 Proofs for center-based clustering

Lemma 3.3.6. *Let $P \subset B_2^d$ be a set of points. Let V be the set of all cost vectors of P for (k,z)-clustering. Then there exists an ε-clustering net of size*

$$|\mathcal{N}(V, \|.\|_\infty, \varepsilon)| \leq \exp(O(1) \cdot z \cdot k \cdot d \cdot \log(z\varepsilon^{-1})).$$

Proof. We start by proving the bound for $k = 1$. Suppose we are given a net $\mathcal{N}(B_2^d, \|.\|_2, \delta)$, for a δ to be determined later. Consider a candidate solution $\{s\}$ with cost vector $v^{\{s\}} \in V$. Let s' be the point in $\in \mathcal{N}(B_2^d, \|.\|_2, \delta)$ of such that $\|s - s'\| \leq \delta$, if s' is not unique any one will be sufficient. Let $v^{\mathcal{S}'}$ be the cost vector of $\mathcal{S}'$. The number of distinct solutions $\mathcal{S}'$ are $|\mathcal{N}(B_2^d, \|.\|_2, \delta)| = \exp(O(1) \cdot d \cdot \log \delta^{-1})$ due to Lemma 3.2.3.

What is left to show is that all solutions constructed in this way satisfy the guarantee of $\mathcal{N}(V, \|.\|_\infty, \delta)$, for an appropriately chosen δ. We have for any $p \in P$ and any non-negative integer z due to Lemma 3.2.4

$$\begin{aligned} \left| \|p - s\|^z - \|p - s'\|^z \right| &\leq \alpha \cdot \|p - s\|^z + \left(\frac{2z + \alpha}{\alpha}\right)^{z-1} \|s - s'\|^z \\ &\leq \alpha \cdot \|p - s\|^z + (3z)^z \left(\frac{\delta}{\alpha}\right)^{z-1} \cdot \delta \end{aligned}$$

We set $\alpha = \frac{1}{2 \cdot 2^z}\varepsilon$ and $\delta = \alpha \cdot \frac{1}{2(3z)^z}\varepsilon = \frac{1}{4(6z)^z}\varepsilon^2$. Then the term above is upper bounded by at most ε as $\|p - s\| \leq 2$. Now since $\left| \|p - s\|^z - \|p - s'\|^z \right| \leq \varepsilon$ for all $s \in B_2^d$ also implies $|\min_{s \in \mathcal{S}} \|p - s\|^z - \min_{s' \in \mathcal{S}'} \|p - s'\|^z| \leq \varepsilon$, we have proven our desired approximation.

To conclude, observe that by our choice of δ, the overall net N has size at most $\exp(O(1) \cdot z \cdot d \cdot \log(z\varepsilon^{-1}))$.

To extend this proof to k-centers, observe that any solution consisting of k centers can be obtained by selecting k points from B_2^d, rather than one. This raises the net size of the single cluster case by a power of k. □

We now show that Lemma 3.3.6 combined with terminal embeddings yields the desired net.

Lemma (Equation 3.10 in Lemma 3.3.2)**.** Let $\mathcal{D}$ be a distribution over B_2^d and let P a set of n points sampled from $\mathcal{D}$ and let V be defined as in Theorem 3.8. Then

$$|\mathcal{N}(V, \|.\|_\infty, \varepsilon)| \leq \exp(O(1)z^3 \cdot k \cdot \varepsilon^{-2} \log n \cdot (\log(z) + \log(\varepsilon^{-1}))).$$

Proof. Let $f : \mathbb{R}^d \to \mathbb{R}^m$ be a terminal embedding, that is f is such that $m \in O(z^2 \cdot \varepsilon^{-2} \log |P|)$[3] and for all $p \in P$ and $q \in \mathbb{R}^d$

$$\|p - q\|^z = (1 \pm \varepsilon)\|f(p) - f(q)\|^z,$$

as given by Narayanan and Nelson (2019). Therefore, for any candidate solution $\mathcal{S}$, we also have

$$\text{cost}(p, \mathcal{S}) = (1 \pm 2\varepsilon)\text{cost}(f(p), f(\mathcal{S})).$$

In other words, the set of cost vectors in the image of f is the desired $O(\varepsilon)$-net for the true set of cost vectors. Hence an ε-net for the cost vectors induced

[3]The dependency on z is easily derived via a straightforward application of Lemma 3.2.4.

by solutions in the image of f is also an $O(\varepsilon)$-net for the set of cost vectors. We thus may apply Lemma 3.3.6 for all cost vectors induced by solutions in the image of f. After rescaling ε by constant factors, the overall net size is therefore $\exp(O(1)z^3 \cdot k \cdot \varepsilon^{-2} \log n \cdot (\log(z) + \log(\varepsilon^{-1})))$

□

3.3.2 Proofs for subspace clustering

In this section, we provide full proofs for Section 3.3 relative to subspace clustering.

We start with a few basic lemmas that will be useful in the calculations later.

We further require the following bounds that will prove useful in the calculations later.

Lemma 3.3.7. *Let a, b be numbers in $[0, 2]$ and let $\varepsilon > 0$. Suppose $a^2 = b^2 \pm \varepsilon \cdot b$. Then*

$$|a - b| \leq \varepsilon.$$

Moreover, for any non-negative integer z, we have

$$|a^z - b^z| \leq 2 \cdot (3z)^z \cdot \varepsilon.$$

Proof. For the first part of the lemma, we observe

$$|a^2 - b^2| = |a - b| \cdot (a + b) \leq \varepsilon \cdot b$$

which implies

$$|a - b| \leq \varepsilon.$$

For the second part, Lemma 3.2.4 implies
$|a^z - b^z| \leq \varepsilon \cdot \max(a, b)^z + \left(\frac{2z + \varepsilon}{\varepsilon}\right)^{z-1} \cdot |a - b|^z \leq \varepsilon \cdot 2^z + \left(\frac{3z + \varepsilon}{\varepsilon}\right)^{z-1} \cdot \varepsilon^z \leq 2(3z)^z \varepsilon.$ □

This lemma now immediately implies the following corollary by rescaling ε.

Corollary 3.3.7.1. *Let a, b be numbers in $[0, 2]$ and let $\varepsilon > 0$. Suppose $a^2 = b^2 \pm \frac{1}{4 \cdot (3z)^z} \max(\varepsilon \cdot b, \varepsilon^2)$. Then for any non-negative integer z, we have*

$$|a^z - b^z| \leq \varepsilon.$$

We now show that for any candidate subspace U we can find a subspace representing it that is spanned by only a few vectors in P.

Lemma (Lemma 3.3.5)**.** Let $P \subseteq B_2^d$. For any orthogonal matrix $U \in \mathbb{R}^{j \times d}$, there exists $M \subseteq P$, with $|M| = O(j \cdot \varepsilon^{-2})$, such that

$$\forall p \in P, \|U^T (I - \Pi_M) p\| \leq \varepsilon \cdot \|(I - \Pi_M) p\|. \tag{3.13}$$

Proof. Initially, let $M = \emptyset$. We add points to M in rounds and denote by M_t the set after t rounds. Furthermore, let Π_t be the projection matrix onto the subspace spanned by M_t at round t. If there is a $p \in P$ in round t such that

$$\|U^T (I - \Pi_t) p\| > \varepsilon \|(I - \Pi_t) p\| \tag{3.14}$$

then we let $M_{t+1} = M_t \cup \{p\}$. Our goal is to show that after $T \in O(j\varepsilon^{-2})$ many rounds, we have $\|U^T(I-\Pi_T)p\| \leq \varepsilon \cdot \|(I-\Pi_T)p\|$. We show this by proving inductively

$$\|U^T \Pi_t\|_F^2 \geq \varepsilon^2 \cdot t.$$

For the base case $t = 0$, this is trivially true. Thus suppose we add a point p in iteration $t+1$. Reformulating Equation 3.14, we have $\frac{\|U^T(I-\Pi_t)p\|}{\|(I-\Pi_t)p\|} > \varepsilon$. By the Pythagorean theorem, we therefore have

$$\|U^T \Pi_{t+1}\|_F^2 = \|U^T \Pi_t\|_F^2 + \frac{\|U^T(I-\Pi_t)p\|^2}{\|(I-\Pi_t)p\|^2} \geq \varepsilon^2 \cdot t + \varepsilon^2 \geq \varepsilon^2 \cdot (t+1).$$

Now since Π_t is a projection and since U has j orthonormal columns $j \geq ||U^T||_F^2 \geq ||U^T \Pi_t||_F^2$. If $T \geq \varepsilon^{-2} j$ we obtain $||U^T \Pi_T||_F^2 \geq j$. This implies that U is contained in the space spanned by M_T. Conversely, U must also be orthogonal to the kernel of M_T that is $U(I - \Pi_T) = 0$. Therefore after at most $\varepsilon^{-2} j$ rounds, we have $\|U^T(I-\Pi_T)p\| \leq \varepsilon \cdot \|(I-\Pi_t)p\|$. □

Lemma (Lemma 3.3.4)**.** Let $P \subset B_2^d$ be a set of points and let z be a positive integer. Then there exists an (ε, j)-projective net of size

$$|\mathcal{N}(V, \|.\|_\infty, \varepsilon)| \leq \exp(O(1) \cdot d \cdot j \cdot \log(j\varepsilon^{-1})).$$

Proof. Let N be an ε/j-net of the Euclidean unit ball, i.e. $N = \mathcal{N}(B_2^d, \|.\|_2, \varepsilon/j)$ due to Lemma 3.2.3. Let $\mathcal{N} = \otimes_{i=1}^j N$ be the set of $j-$subsets of of N. We claim that for every S, there exists an $S' \in \mathcal{N}$ such that

$$\|S^T p\|_2 = \|S'^T p\|_2 \pm \varepsilon.$$

Note that this implies the claim as $|\mathcal{N}| \in \left(\left(1 + \frac{2j}{\varepsilon}\right)^d\right)^j = \exp(O(1) \cdot d \cdot j \cdot \log(j\varepsilon^{-1}))$.

Define $S_i'^T$ to be the vector in N closest to the ith row of S^T, i.e. $\|S_i^T - S_i'^T\|_2 \leq \varepsilon/j$. We have $\|S'^T - S\|_2 \leq \sum_{i=1}^j \|S_i'^T - S_i^T\|_2 \leq \varepsilon$. Therefore

$$\begin{aligned}\|S^T p\|_2 &= \|(S^T - S'^T)p + S'^T p\|_2 \\ &\leq \|(S^T - S'^T)p\|_2 + \|S'^T p\|_2 \\ &\leq \|S'^T p\|_2 + \|S^T - S'^T\|_2 \|p\|_2 \\ &\leq \|S'^T p\|_2 + \varepsilon.\end{aligned}$$

The bound $\|S^T p\|_2 \geq \|S'^T p\|_2 - \varepsilon$ is proven analogously. □

We can now conclude with the proof of Equation 3.11 in Lemma 3.3.2.

Lemma (Equation 3.11 in Lemma 3.3.2)**.** Let $\mathcal{D}$ be a distribution over B_2^d and let P a set of n points sampled from $\mathcal{D}$ and let $V_{j,z}$ be defined as in Theorem 3.3.1. Then

$$|\mathcal{N}(V_{j,z}, \|.\|_\infty, \varepsilon)| \leq \exp(O(1)(3z)^{z+2} \cdot k \cdot \varepsilon^{-2}(\log n + j \log(j\varepsilon^{-1})) \log \varepsilon^{-1}).$$

Proof. Let $\alpha, \beta > 0$ be sufficiently small parameters depending on ε that will determined later. We first describe a construction for nets for a single subspace of rank at most j, before composing to k subspaces.

We start by describing the composition of the nets. For every subset $M \subseteq P$, with $|M| \in O(j\alpha^{-2})$, we let Π_M denote an orthogonal projection matrix of the span of M. Note that this implies $\mathbf{rank}(\Pi_M) = O(j\alpha^{-2})$. Further, let $N(\Pi_M) := \mathcal{N}(B_2^{\mathbf{rank}(\Pi_M)}, \|.\|_2, \beta)$ be a (β, j)-projective net of the point set $\cup_{p\in M}\{\Pi_M p\}$ of size at most $\exp(O(1) \cdot \mathbf{rank}(\Pi_M) \cdot \log(j\beta^{-1}))$ given by Lemma 3.3.4. Finally, let $N := \cup_M N(\Pi_M)$.

We consider an arbitrary orthogonal matrix $U \in \mathbb{R}^{j\times d}$. Denote by M_U the subset of points and by Π_U the projection matrix given by Lemma 3.3.5, using α as the precision variable. We claim that for every U, there exists an $U' \in N$ such that for all $p \in P$

$$\left|\left(\|\Pi_U p\|_2^2 - \|U'^T \Pi_U p\|_2^2 + \|(I-\Pi_U)p\|_2^2\right)^{z/2} - \|(I-UU^T)p\|^z\right| \in O(\alpha+\beta).$$

In other words, by enumerating over all (β, j)-projective nets, we obtain an $O(\alpha+\beta)$-subspace clustering net for $(1, j, z)$-clustering. The desired error of ε then follows by choosing α and β accordingly. For U, we construct U' as follows. Let $D = \sqrt{\Pi_U}$, i.e. $DD^T = \Pi_U$. Further, let $V = U^T D$, notice that V has at most j rows that have at most unit norm. Hence, there exists a $U' \in N$ such that

$$|\|U\Pi_U p\|_2 - \|U'\Pi_U p\|_2| \leq \varepsilon$$

that is a (β, j)-projective net.

We then obtain

$$\begin{array}{rcl} & & \|\Pi_U p\|_2^2 - \|U'^T \Pi_U p\|_2^2 + \|(I-\Pi_U)p\|_2^2 \\ & = & \|\Pi_U p\|_2^2 - \|U^T \Pi_U p\|_2^2 \pm \beta + \|(I-\Pi_U)p\|_2^2 \\ & = & \|\Pi_U p\|_2^2 - \|UU^T \Pi_U p\|_2^2 \pm \beta + \|(I-\Pi_U)p\|_2^2 \\ & = & \|(I-UU^T)\Pi_U p\|_2^2 + \|(I-\Pi_U)p\|_2^2 \pm \beta \\ (Eq.3.12) & = & \|(I-UU^T)p\|_2^2 \pm \beta - \|U^T(I-\Pi_U)p\|^2 - 2p^T\Pi_U UU^T(I-\Pi_U)^T p \\ (Lem.3.3.5) & = & \|(I-UU^T)p\|_2^2 \pm \alpha^2 \cdot \|(I-UU^T)p\|^2 \pm 2\alpha \cdot \|(I-UU^T)p\| \pm \beta \end{array}$$

Setting $\alpha^2 = \beta = \frac{1}{64(3z)^z}\varepsilon^2$, we then have due to Corollary 3.3.7.1

$$\left|\left|\|\Pi_U p\|_2^2 - \|U'^T \Pi_U p\|_2^2 + \|(I-\Pi_U)p\|_2^2\right|^z - \|(I-UU^T)p\|^z\right| \leq \varepsilon. \quad (3.15)$$

To extend this from a single j-dimensional subspace to a solution $\mathcal{U}$ given by the intersection of k j-dimensional subspaces, we define cost vectors $v^{\mathcal{S}'}$ obtained from $\mathcal{N} = \otimes_{i=1}^k N$ as follows. For each $U \in \mathcal{U}$ let U' be constructed as above and let $\mathcal{U}'$ be the union of the thus constructed U'. Then, with a slight abuse of notation, letting $\Pi_{U'}$ correspond to the subspace used to obtain U', we define

$$v_p^{\mathcal{U}'} := \min_{U'\in\mathcal{U}'} \left|\|(I-\Pi_{U'})p\|^2 + \|\Pi_{U'}p\|^2 - \|U'\Pi_{U'}p\|^2\|\right|^{z/2}.$$

Let U be the subspace to which p is assigned $\mathcal{U}$ and let U' be the center in $\mathcal{U}'$ used to approximate U and let $U^{*\prime} = \text{argmin}_{U'\in\mathcal{U}'} |\|(I-\Pi_U')p\|^2 + \|\Pi_{U'}p\|^2 - \|U'\Pi_U' p\|^2\||^{z/2}$

and let $U^* \in \mathcal{U}$ be the center approximated by $U^{*\prime}$. Then applying Equation 3.15, we have

$$\begin{aligned} & \|(I - UU^T)p\|^z \\ \leq\ & \|(I - U^*U^{*T}p\|^z \\ \leq\ & \left| \|(I - \Pi_{U^{*\prime}})p\|^2 + \|\Pi_{U^{*\prime}}p\|^2 - \|U*'\Pi_{U^{*\prime}}p\|^2 \right|^{z/2} + \varepsilon \\ \leq\ & \left| \|(I - \Pi_{U'})p\|^2 + \|\Pi_{U'}p\|^2 - \|U'\Pi_{U'}p\|^2 \right|^{z/2} + \varepsilon \end{aligned}$$

Thus, the cost vectors obtained from $\mathcal{N}$ are a (k, j, z)-clustering net, i.e.

$$\left| v_p^{\mathcal{S}'} - v_p^{\mathcal{S}} \right| := \left| \min_{s' \in \mathcal{S}'} \|\Pi_{s'}p - [s', 0]\|^z - \min_{s \in \mathcal{S}} \|p - s\|^z \right| \leq \varepsilon.$$

What remains is to bound the size of the clustering net. Here we first observe that size of the clustering net is equal to $|\mathcal{N}| = |N|^k$. For $|N|$, we have $\binom{|P|}{O(\alpha^{-2}\log\alpha^{-1})} \leq n^{O(j\alpha^{-2}\log\alpha^{-1})}$ many choices of $N(\Pi)$. In turn, the size of each $N(\Pi)$ is bounded by $(\beta/j)^{-O(j^2\alpha^{-2})}$ due to Lemma 3.3.4. Thus the overall size of $\mathcal{N}$ is

$$\begin{aligned} & \exp\left(k \cdot j \cdot O(\alpha^{-2}\log\alpha^{-1}(\log n + j\log\beta/j))\right) \\ = & \exp(O(1)(3z)^{z+2} \cdot k \cdot j \cdot \varepsilon^{-2}(\log n + j\log(j\varepsilon^{-1}))\log\varepsilon^{-1}) \end{aligned}$$

as desired. □

Proofs of Theorem 3.4.1 (Section 3.4)

The proof of the theorem is a straightforward application of Theorem 3.2.1 with the following Lemma

Lemma 3.3.8. *Let $\mathcal{D}$ be a distribution over B_2^d, let P a set of n points sampled from $\mathcal{D}$, and let V be defined as in Theorem 3.4.1. Then for any $\gamma > 0$*

$$Rad_n(V_{j,2}) \in O\left(\sqrt{\frac{kj}{n}\log^{3+\gamma}\left(\frac{n}{j}\right)}\right).$$

Proof. We use the following result due to Foster and Rakhlin (2019).

Theorem 3.3.9 (ℓ_∞ contraction inequality (Theorem 1 by Foster and Rakhlin (2019))). *Let $F \subseteq X \to \mathbb{R}^k$, and let $\phi : \mathbb{R}^k \to \mathbb{R}$ be L-Lipschitz with respect to the ℓ_∞ norm, i.e. $\|\phi(X) - \phi(X')\|_\infty \leq L \cdot \|X - X'\|_\infty$ for all $X, X' \in \mathbb{R}^k$. For any $\gamma > 0$, there exists a constant $C > 0$ such that if $|\phi_t(f(x))| \vee \|f(x)\|_\infty \leq \beta$, then*

$$Rad_n(\phi \circ F) \leq C \cdot L\sqrt{K} \cdot \max_i Rad_n(F|_i) \cdot \log^{3/2+\gamma}\left(\frac{\beta n}{\max_i R_n(F|_i)}\right).$$

We use this theorem as follows. Our functions are associated with candidate solutions $\mathcal{U}$, that is $\phi(f) = \min_{U \in \mathcal{U}} \|(I - UU^T)p\|_2^2$. In other words, f maps a point p to the k-dimensional vector, where $f_i(p) = \|(I - U_iU_i^T)p\|_2^2$ and ϕ selects the minimum value among all $\|I - U_iU_i^T)p\|_2^2$.

Thus, we require three more steps. First, we have to bound the Lipschitz constant of the minimum operator. Second, we have to give a bound on β. Third and last, we have to give a bound on the Rademacher complexity

$$Rad_n(V) = \frac{1}{n} \cdot \mathbb{E}_r \sup_U \sum_{p\in P} \|(I - UU^T)p\|_2^2 r_p. \tag{3.16}$$

The Lipschitz constant of the minimum operator with respect to the ℓ_∞ norm can be readily shown to be 1 as for any two vectors x, y with $\min_i y_i = y_j$

$$\min_i x_i - \min_i y_i = \min_i x_i - y_j \leq x_j - y_j \leq |x_j - y_j| \leq \|x - y\|_\infty.$$

Since U is an orthogonal matrices and $p \in B_2^d$, we have $\|(I - UU^T)p\|_2^2 \leq 1$ and thus β is bounded by 1.

Thus, we only require a bound on Equation 3.16. For this, we use a result by Lauer (2020). Since the result is embedded in the proof of another result, we restate it here for the convenience of the reader.

Lemma 3.3.10 (Compare the proof Theorem 3 of Lauer (2020))**.** *Let P be an set of n points in B_2^d and let $\mathcal{U}$ be the set of all orthogonal matrices of rank at most j. For every $U \in \mathcal{U}$, define $f_U(p) = \|(I - UU^T)p\|_2^2$ and let F be the set of all functions $f_U(p)$ Then.*

$$Rad_n(F) := \frac{1}{n} \cdot \mathbb{E}_r \sup_{U\in\mathcal{U}} \sum_{p\in P} \|(I - UU^T)p\|_2^2 \cdot r_p \in O\left(\sqrt{\frac{j}{n}}\right).$$

Proof. We have

$$Rad_n(F) = \mathbb{E}_r \sup_U \sum_{p\in P} \|(I - UU^T)p\|_2^2 r_p = \mathbb{E}_r \sum_{p\in P} \|p\|^2 r_p + \mathbb{E}_r \sup_U \sum_{p\in P} \|U^T p\|_2^2 r_p.$$

We observe that the term $\mathbb{E}_r \sum_{p\in P} \|p\|^2 r_p$ is 0. Thus, we focus on the second term. We have

$$\begin{aligned}
\mathbb{E}_r \sup_U \sum_{p\in P} \|U^T p\|_2^2 \cdot r_p &= \mathbb{E}_r \sup_U \sum_{p\in P} p^T UU^T p \cdot r_p = \mathbb{E}_r \sup_U \sum_{p\in P} trace(p^T UU^T p) \cdot r_p \\
&= \mathbb{E}_r \sup_U \sum_{p\in P} trace(UU^T pp^T) \cdot r_p \\
&= \mathbb{E}_r \sup_U trace\left(UU^T \sum_{p\in P} \left(r_p \cdot pp^T\right)\right) \\
&\leq \mathbb{E}_r \sup_U \|U\|_F \left\|\sum_{p\in P} r_p \cdot pp^T\right\|_F.
\end{aligned}$$

We have $\|U\|_F \leq \sqrt{j}$, so we focus on $\left\|\sum_{p\in P} r_p \cdot pp^T\right\|_F$. Here, we have

$$\begin{aligned}
\left\|\sum_{p\in P} r_p \cdot pp^T\right\|_F^2 &= trace\left(\left(\sum_{p\in P} r_p \cdot pp^T\right)\left(\sum_{p\in P} r_p \cdot pp^T\right)\right) \\
&= \sum_{p\in P}\sum_{q\in P} r_p \cdot r_q \cdot trace\left(pp^T qq^T\right) = \sum_{p\in P}\sum_{q\in P} r_p \cdot r_q \cdot (p^T q)^2.
\end{aligned}$$

This implies

$$\begin{aligned}
n \cdot Rad_n(F) &= \mathbb{E}_r \sup_U \sum_{p\in P} \|U^T p\|_2^2 r_p \le \mathbb{E}_r \sup_U \|U\|_F \left\| \sum_{p\in P} r_p \cdot pp^T \right\|_F \\
&\le \sqrt{j} \cdot \mathbb{E}_r \sqrt{\sum_{p\in P}\sum_{q\in P} r_p \cdot r_q \cdot (p^T q)^2} \\
\text{(Jensen's inequality)} &\le \sqrt{j} \cdot \sqrt{\mathbb{E}_r \sum_{p\in P}\sum_{q\in P} r_p \cdot r_q \cdot (p^T q)^2} \\
&= \sqrt{j} \cdot \sqrt{\sum_{p\in P} (p^T p)^2} \le \sqrt{j} \cdot \sqrt{\sum_{p\in P} 1} = \sqrt{nj}.
\end{aligned}$$

Solving the above for $Rad_n(F)$ concludes the proof. □

We can now conclude the proof. Combining the bounds on L and β with Lemma 3.3.10 and Theorem 3.3.9, we have

$$Rad_n(V_{j,2}) \in O\left(\sqrt{k} \cdot \sqrt{\frac{j}{n}} \cdot \log^2(n)\right)$$

as desired. □

3.4 Tight generalization gounds for projective clustering

For the specific case of $(k, j, 2)$ clustering, also known as projective clustering, we obtain an even better dependency on j. A similar bound can likely also be derived using the seminal work of Fefferman et al. (2016), though the dependencies on $\log n$ and $\log 1/\delta$ are slightly weaker. The proof uses the main result by Foster and Rakhlin (2019), itself heavily inspired by Fefferman et al. (2016), and arguments related to bounding the Rademacher complexity of linear function classes. Crucially, it avoids the issue of obtaining an explicit dimension reduction entirely, but the approach cannot be extended to general (k, j, z) clustering.

Theorem 3.4.1. *Let $\mathcal{D}$ be a distribution over B_2^d and let P a set of n points sampled from $\mathcal{D}$. For any set $\mathcal{U}$ of k orthogonal matrices of rank at most j, we denote by $v^{\mathcal{U}}$ the n-dimensional cost vector of P in solution $\mathcal{U}$ with respect to the $(k, j, 2)$-clustering objective, i.e. $v_p^{\mathcal{U}} = \min_{U\in\mathcal{U}} \|(I - UU^T)p\|^2$. Let $V_{j,2}$ be the union of all cost vectors of P. Then with probability at least $1-\delta$ for any $\gamma > 0$*

$$\mathcal{E}_n(V_{j,2}) \in O\left(\sqrt{\frac{kj}{n} \cdot \log^{3+\gamma}\left(\frac{n}{j}\right)} + \sqrt{\frac{\log 1/\delta}{n}}\right).$$

Finally, we also show that the bounds from Theorem 3.4.1 and Fefferman et al. (2016) are essentially optimal.

Theorem 3.4.2. *There exists a distribution $\mathcal{D}$ supported on B_2^d such that $\mathcal{E}_n(V_{j,2}) \in \Omega\left(\sqrt{(kj)/n}\right)$.*

The rough idea is to define a distribution $\mathcal{D}$ supported on the nodes of a $2kj$-dimensional simplex with some points having more probability mass and some points having smaller mass. Using the tightness of Chernoff bounds, we may then show that the probability of fitting a subspace clustering to a good fraction of the lower mass points is always sufficiently large.

3.4.1 Proofs for the lower bound

Finally, we also show that the bound given in Theorem 3.4.1 is optimal, up to polylog factors.

Theorem (3.4.2)**.** There exists a distribution $\mathcal{D}$ supported on B_2^d such that $\mathcal{E}(V_{j,2}) \in \Omega\left(\sqrt{\frac{kj}{n}}\right)$.

Proof. We first describe the hard instance distribution $\mathcal{D}$. We assume that we are given $d = 2kj$ dimensions. Let e_i be the standard unit vector along dimension i with $i \in \{1, \dots d\}$. Let $p, \varepsilon \in [0,1]$ be a parameters, where ε is sufficiently small. We set the densities for a point q as follows.

$$\mathbb{P}[q] = \begin{cases} p & \text{if } q = e_i, i \in \{1, \dots, k \cdot j\} \\ p - \varepsilon \cdot p & \text{if } q = e_i, i \in \{kj+1, \dots, d\} \\ 0 & \text{otherwise} \end{cases} \tag{3.17}$$

We choose p such that integral over densities is 1, i.e. $kj \cdot p + kj \cdot (p - \varepsilon p) = 1$. It is straightforward to verify that for ε sufficiently small, $p \in (\frac{1}{kj}, \frac{2}{kj})$. We denote the points $\{e_1, \dots e_{kj}\}$ by G for "good" and the points $\{e_{kj+1}, \dots e_d\}$ by B for "bad".

We now characterize the properties of the optimal solution as well as suboptimal solutions.

Lemma 3.4.3. *Let $\mathcal{D}$ be the distribution described above in Equation (3.17). Then for any optimal solution $\mathcal{U} = \{U_1, \dots U_k\}$, we have $e_i \in U_t$ for $i \in \{1, \dots, kj\}$ and some t and $OPT = kj \cdot p \cdot (1 - \varepsilon)$.*

Proof. We transform the instance into a $d \times d$ diagonal matrix D where $D_{i,i} = \sqrt{\mathbb{P}[e_i]}$. So D is a $d \times d$ diagonal matrix with diagonal entries equal to $\sqrt{p}$ for the first $k \cdot j$ elements and $\sqrt{p - \varepsilon \cdot p}$ for elements from $k \cdot j + 1$ to d. Now consider any partition of the points into clusters C_t with the corresponding subspace U_t for ($t \in \{1, \dots, k\}$). The optimal solution for U_t is simply the right singular vector of the submatrix of D corresponding to points in C_t, which by the construction of D is the j points with the largest weight. This means that each cluster can remove at most $\sum_{i=1}^{j} 1 = j$ from the cost, so k clusters can remove at most $\sum_{i=1}^{k} j$ from the cost. This imples that the cost of the clustering is lower bounded by $\sum_{i=1}^{d} D_{i,i}^2 - \sum_{i=1}^{kj} D_{i,i}^2 = \sum_{i=kj+1}^{d} D_{i,i}^2$. Conversely, the solution $\mathcal{U}$ has exactly this cost, which implies that it must be optimal. □

Using Lemma 3.4.3, we now have to, given n independent samples from $\mathcal{D}$. Control the probability that the sample P will (falsely) put a higher weight on some of the points in B than the points in G. Let B_{ex} denote the set of misclassified points in B and let P_{OPT} denote the optimum computed on the sample P. We have

$$\mathbb{E}[\text{cost}(\mathcal{D}, P_{\text{OPT}})] = kj \cdot p \cdot (1 - \varepsilon) + p \cdot \varepsilon \cdot |B_{ex}|.$$

and hence an expected excess risk bound of

$$\mathbb{E}[\mathrm{cost}(\mathcal{D}, P_{\mathrm{OPT}})] - \mathrm{OPT} = p \cdot \varepsilon \cdot \mathbb{E}[B_{ex}].$$

By linearity of expectation, we have $\mathbb{E}[|B_{ex}|] = kj \cdot \mathbb{P}[e_{kj+1} \in B_{ex}]$. Thus, $\mathbb{E}[\mathrm{cost}(\mathcal{D}, P_{\mathrm{OPT}})] - \mathrm{OPT} \in \Theta(1)\varepsilon \cdot \mathbb{P}[e_{kj+1} \in B_{ex}]$. Define G_{low} to be the set of points from G that are have an empirical density of at most p. Let $\widehat{e_{kj+1}}$ denote the empirical density of e_{kj+1}. We now claim that

$$\begin{aligned}\mathbb{P}[e_{kj} \in B_{ex}] &\geq \mathbb{P}[\widehat{e_{kj+1}} > p \wedge e_{kj+1} \in B_{ex}] \\ &= \mathbb{P}[e_{kj+1} \in B_{ex} | \widehat{e_{kj+1}} > p] \cdot \mathbb{P}[\widehat{e_{kj+1}} > p] \geq 1/2 \cdot \mathbb{P}[\widehat{e_{kj+1}} > p]\end{aligned}$$

The first inequality follows because we are considering a subset of the possible events, the second inequality follows because the expected number of points with an empirical estimated density greater than p is negatively correlated with the empirical density $\widehat{e_{kj+1}}$ of the point e_{kj}. Specifically, conditioned on $\widehat{e_{kj+1}} > p$, the median density of any point $e_i \in G$ is at most $\frac{1}{n} \cdot p(n - p \cdot n) = p \cdot (1-p) < p$. Thus, the (marginal) expected density of any other point is below p and therefore the probability that e_{kj+1} will be in B_{ex} is at least $1/2$.

Thus, what remains to be shown is a bound on $\mathbb{P}[e_{kj} > p]$. Here, we use the tightness of the Chernoff bound (see Lemma 4 of Klein and Young (2015)).

Lemma 3.4.4 (Tightness of the Chernoff Bound)**.** *Let X be the average of n independent, $0/1$ random variables. For any $\varepsilon \in (0, 1/2]$ and $\mu \in (0, 1/2]$, assuming $\varepsilon^2 \mu n \geq 3$ if each random variable is 1 with probability at least μ, then*

$$\mathbb{P}[X > (1+\varepsilon)p] > \exp(-9\varepsilon^2 \mu n).$$

Thus, sampling n elements, we have

$$\begin{aligned}\mathbb{P}\left[e_{kj} > p\right] &= \mathbb{P}\left[e_{kj} - (1-\varepsilon)p > \frac{\varepsilon}{1-\varepsilon} \cdot (1-\varepsilon) \cdot p\right] \\ &> \exp\left(-9\frac{\varepsilon^2}{(1-\varepsilon)^2}(1-\varepsilon)pn\right) \in \Omega(1)\exp\left(-\frac{\varepsilon^2}{kj}n\right).\end{aligned}$$

If we require $\mathbb{E}[\mathrm{cost}(\mathcal{D}, P_{\mathrm{OPT}})] - \mathrm{OPT} = \varepsilon \cdot c$ for a sufficiently small absolute constant c, we also require $\mathbb{P}\left[e_{kj} > p\right] = c'$ and hence $\sqrt{\frac{kj}{n}} \leq \varepsilon \cdot c''$ for a sufficiently small absolute constants c' and c''. Letting $\varepsilon \to 0$ then shows that the excess risk can asymptotically decrease no faster than $\Omega\left(\sqrt{\frac{kj}{n}}\right)$. □

3.5 Experiments

Theoretical guarantees are often notoriously conservative compared to what is seen in practice. In this section, we present empirical findings detailing whether the risk bounds from the previous sections are also the risk bounds one can expect when dealing with real datasets. Generally, two properties can determine the risk decrease. First, the clusters may be well separated (Angelidakis et al., 2017; Cohen-Addad and Schwiegelshohn, 2017). Indeed, making assumptions to this end, there is also some theoretical evidence that a rate of $O(k/n)$ is possible (Antos et al., 2005; Levrard, 2015). The other, somewhat related explanation is that if the ground truth consists of

$k' < k$ clusters (Bhattacharyya et al., 2022; Ostrovsky et al., 2012), the dependency on k will point more towards the smaller, true number of clusters. We run the experiments both for center based clustering, as well as subspace clustering. While the focus of the work is arguably more on subspace clustering, the experiments are important in both cases. Although both problems are hard to optimize exactly, center based clustering is significantly more tractable and thus may lend better insight into practical learning rates. For example, we have an abundance of approximation algorithms for (k, z) clustering (Arthur and Vassilvitskii, 2007; Mettu and Plaxton, 2004) whereas, even in the case of $(k, 1, z)$ clustering in two dimensions (Kumar et al., 2000) it is not possible to find any finite approximation in polynomial time.

Datasets We use four publicly available real-world datasets: Mushroom (Schlimmer, 1987), Skin-Nonskin (Bhatt and Dhall, 2012), MNIST (Lecun et al., 1998), and Covtype (Blackard, 1998). Mushroom comprises of 112 categorical features of the appearance of mushrooms with class labels corresponding to poisonous or edible. MNIST contains 28x28 pixel images of handwritten digits. Skin_Nonskin are RGB values given as 3 numerical features used to predict if a pixel is skin or not. Lastly, Covtype consists of a mix of categorical and numerical features used to predict seven different cover types of forests. Each dataset was normalized by the diameter, ensuring that all points lie in B_2^d.

Table 3.1. Datasets used for the experiments

Dataset	Points	Dim	Labels
Mushrooms	8,124	112	2
MNIST	60,000	784	10
Skin_Nonskin	245,057	3	2
Covtype	581,012	54	7

Problem parameters and algorithms For both center based clustering as well as subspace clustering, we focus on the powers $z \in \{1, 2, 3, 4\}$. $z = 2$ is arguably the most popular and also the most tractable variant. $z = 1$ is the objective with the least susceptibility to outliers. Finally, we consider the cases $z = 3$, due to it minimizing asymmetry and $z = 4$ as a tractable alternative to the coverage objective $z \to \infty$. The excess risk is evaluated for $k \in \{10, 20, 30, 50\}$ for both center based and subspace clustering. Expectation maximization (EM) type algorithms are used for both center-based and subspace clustering. Given a solution $\mathcal{S}$ we first assign every point to its closest center and subsequently recompute the center. For the case $z = 2$, we do this analytically and in this case the EM algorithm is more commonly known as Llyod's method (Lloyd, 1982). For the cases, $z \in \{1, 3, 4\}$, the new center is obtained via gradient descent. The initial centers are chosen via D^z sampling, i.e. sampling centers proportionate to the zth power of the distance between a point and its closest center (for $z = 2$ this is the k-means++ algorithm by Arthur and Vassilvitskii (2007)).

For subspace clustering, we consider $j \in \{1, 2, 5\}$ to demonstrate the effects of the subspace dimension on convergence rate, taking computational expenses into consideration. Since there are no known tractable algorithms for these problems with guarantees, we initialize a solution $\mathcal{U} = \{U_1, \ldots, U_k\}$ by sampling k orthogonal matrices of rank j, where the subspace for each matrix is determined via the volume sampling technique (Deshpande and Varadarajan, 2007). Subsequently, we run

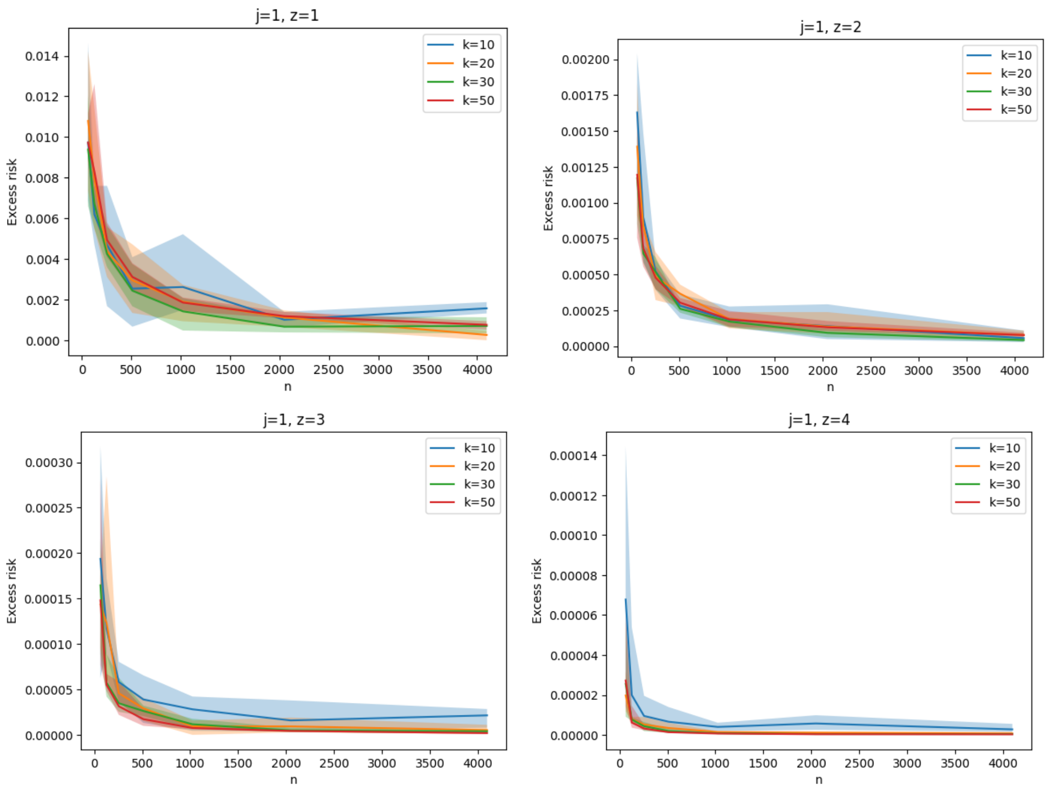


Figure 3.1. Excess risk for line clustering on Covtyp. Shaded areas show max-min intervals.

the EM algorithm. As before, the expectation step consists of finding the closest subspace for every point. For $z = 2$, the maximization step consists of finding the j principal component vectors of the data matrix induced by each cluster. For the other values of z, it is NP-hard even approximate the maximization step (Clarkson and Woodruff, 2015), so we use gradient descent to find a local optimum. Due to the fact that Skin_nonskin only has 3 features, we only evaluate the excess risk for $j \in \{1, 2\}$. Due to a large computational dependency on dimension, we do not evaluate subspaces on the MNIST dataset.

We wrote all of the code using Python 3 and utilized the Pytorch library for implementations using gradient descent. Specifically, we employed the AdamW optimizer to find the closest center with a learning rate set to 0.01. All experiments were conducted on a machine equipped with a single NVIDIA RTX 2080 GPU.

Experimental setup and results To estimate the optimal cost OPT for the two objective functions, we run the corresponding appropriate algorithms mentioned above ten times on the entire dataset P and use the minimal objective value as an estimate for OPT. We obtain a sample S_i of size n by sampling uniformly at random and estimate the optimal cost for that sample, OPT_i. We repeat this 5 times. The empirical excess risk is calculated as $\mathcal{E}_n = \frac{1}{|P|} \sum_{i=1}^{5} \frac{\text{cost}(P, OPT_i)}{5} - OPT$. The excess risk for center-based clustering is evaluated on exponential-sized subset sizes $n \in \{2^6, 2^7, \ldots, 2^{12}\}$.

We fit a line of the form $c \cdot \frac{k^{q_1}}{n^{q_2}}$ where c, q_1, q_2 are the optimizeable parameters. Let y_i be the excess risk in run i. Let k_i and n_i be the values of k and n in run i and let r be the total number of times the excess risk was evaluated for each combination of algorithm and dataset. We use gradient descent on the following loss to optimize the parameters $LSE = \sum_{i=1}^{r} \left(y_i - c \cdot \frac{k^{q_1}}{n^{q_2}} \right)^2$.

The results in Figure 3.1 show that the excess risk for subspace clustering decreases quicker for higher values of z, and we see a similar pattern for center-based clustering. The best-fit lines shown in Tables 3.2 and 3.3 indicate that the empirical excess risk values decrease slightly quicker than predicated by theory. The expected values are $q_1 = q_2 = 0.5$ and we observe q_1, q_2 around $0.44, 0.52$ respectively. For k this indicates a slightly favorable dependency in practice. For q_2, we consider the difference to the theoretical bound of 0.5 negligible. The choice of z does not seem to have a significant impact on either finding. For subspace clustering, the dependency on k is a bit more pronounced and increases slightly towards the theoretical guarantees.

Contrary to hopes that margin or stability conditions might occur on practical datasets, the results indicate that the theoretical guarantees of the learning rate are near-optimal even in practice. Moreover, the rates were not particularly affected by either the choice of z or by the dimension j when analyzing subspace clustering.

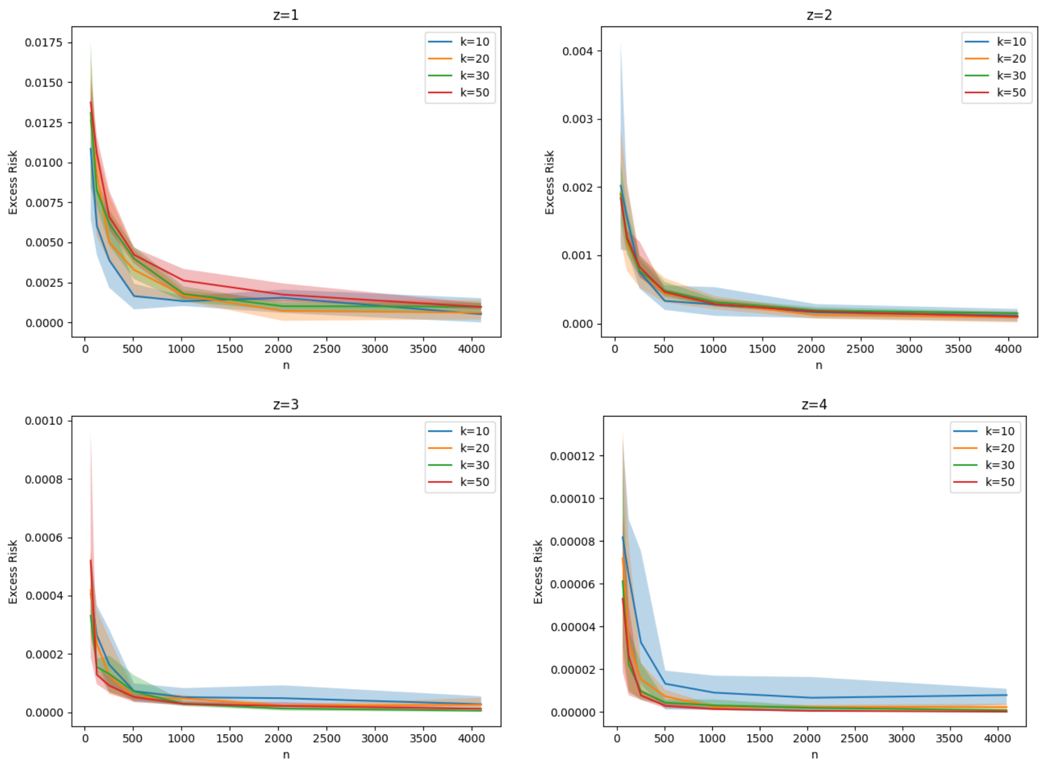


Figure 3.2. Excess risk for center-based clustering on the Covertype dataset. The shaded areas indicate the maximal and minimal deviation for the respective sample sizes.

3.6 Conclusion and open problems

In this work, we presented several new generalization bounds for clustering objectives such as k-median and subspace clustering. When the centers are points

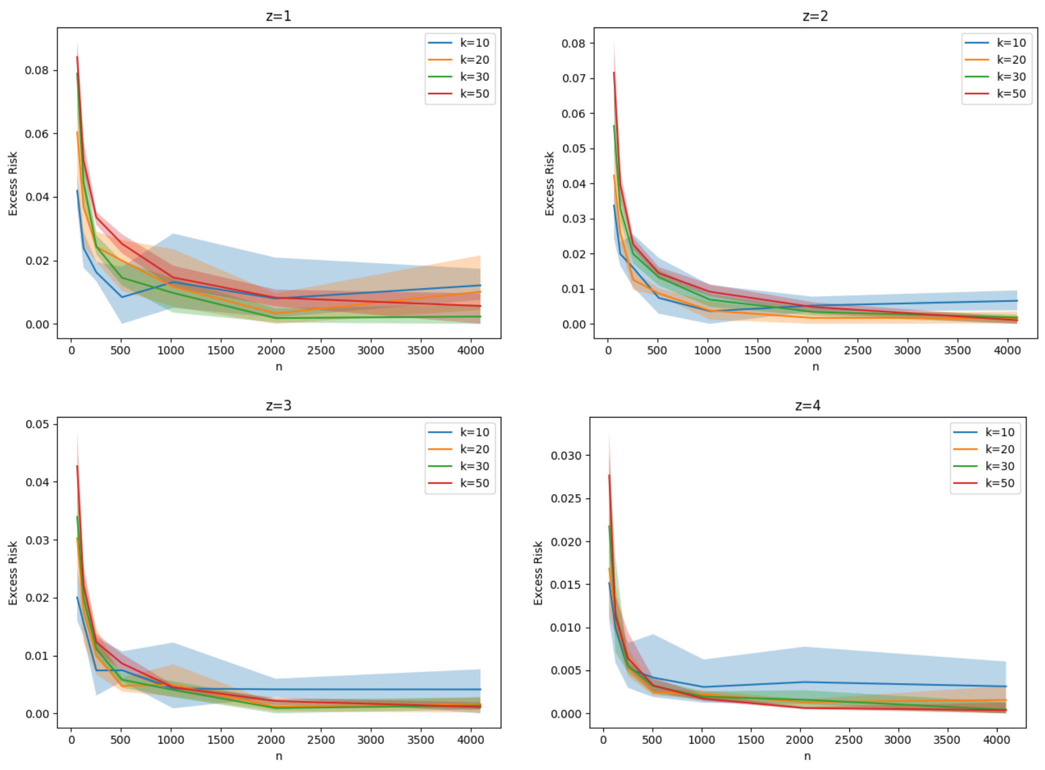


Figure 3.3. Excess risk for center-based clustering on the Mushroom dataset. The shaded areas indicate the maximal and minimal deviation for the respective sample sizes.

Table 3.2. Best fit lines on Covtype and Mushroom (left to right)

z	c	q_1	q_2
1	$3 \cdot 10^{-2}$	0.44	0.54
2	$4 \cdot 10^{-3}$	0.42	0.52
3	$6 \cdot 10^{-4}$	0.44	0.51
4	$1 \cdot 10^{-4}$	0.44	0.51

z	c	q_1	q_2
1	$1 \cdot 10^{-1}$	0.48	0.51
2	$8 \cdot 10^{-2}$	0.48	0.51
3	$4 \cdot 10^{-2}$	0.49	0.50
4	$3 \cdot 10^{-2}$	0.49	0.50

or constant dimensional subspaces, our upper bounds are optimal up to logarithmic terms. For projective clustering, we give a lower bound showing that the results obtained by Fefferman et al. (2016) are nearly optimal. A key novel technique was using an ensemble of dimension reduction methods with very strong guarantees.

An immediate open question is to which degree ensembles of dimension reductions can improve learning rates over a single dimension reduction. Is it possible to find natural problems where there is a separation between the embeddability and the learnablity of a class of problems, or given the ensemble, is it always possible to find a single dimension reduction with the guarantees of the ensemble? Another open question is motivated by the recent treatment of clustering through the lens of computational social choice (Chierichetti et al., 2017). Using current techniques from coresets (Braverman et al., 2022) and learning theory (Foster and Rakhlin, 2019), it seems difficult to improve over the learning rate of $O\left(\sqrt{k^2/n}\right)$ for the fair clustering problem specifically. It it possible to match the bounds for unconstrained

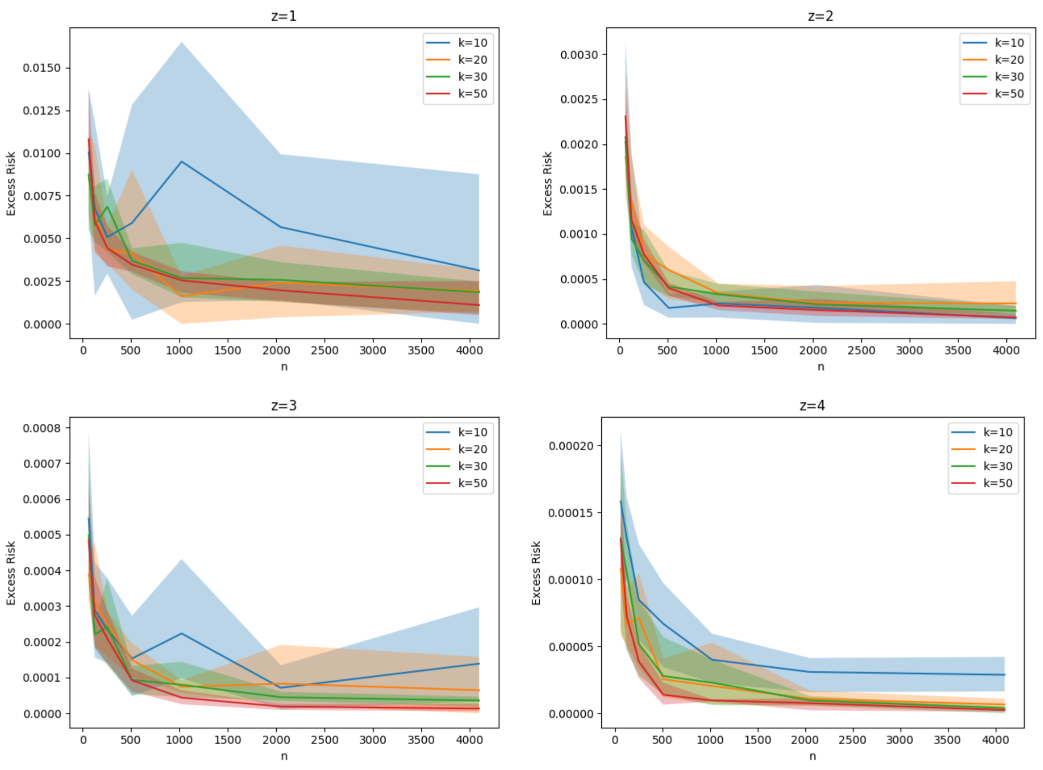


Figure 3.4. Excess risk for center-based clustering on the Skin_Nonskin dataset. The shaded areas indicate the maximal and minimal deviation for the respective sample sizes.

Table 3.3. Best fit lines on Skin_NonSkin and MNIST (left to right)

z	c	q_1	q_2
1	$2 \cdot 10^{-2}$	0.49	0.50
2	$3 \cdot 10^{-3}$	0.47	0.52
3	$8 \cdot 10^{-4}$	0.46	0.53
4	$2 \cdot 10^{-4}$	0.46	0.53

z	c	q_1	q_2
1	$1 \cdot 10^{-1}$	0.49	0.51
3	$5 \cdot 10^{-2}$	0.50	0.50
4	$3 \cdot 10^{-2}$	0.50	0.50
2	$8 \cdot 10^{-02}$	0.50	0.50

clustering?

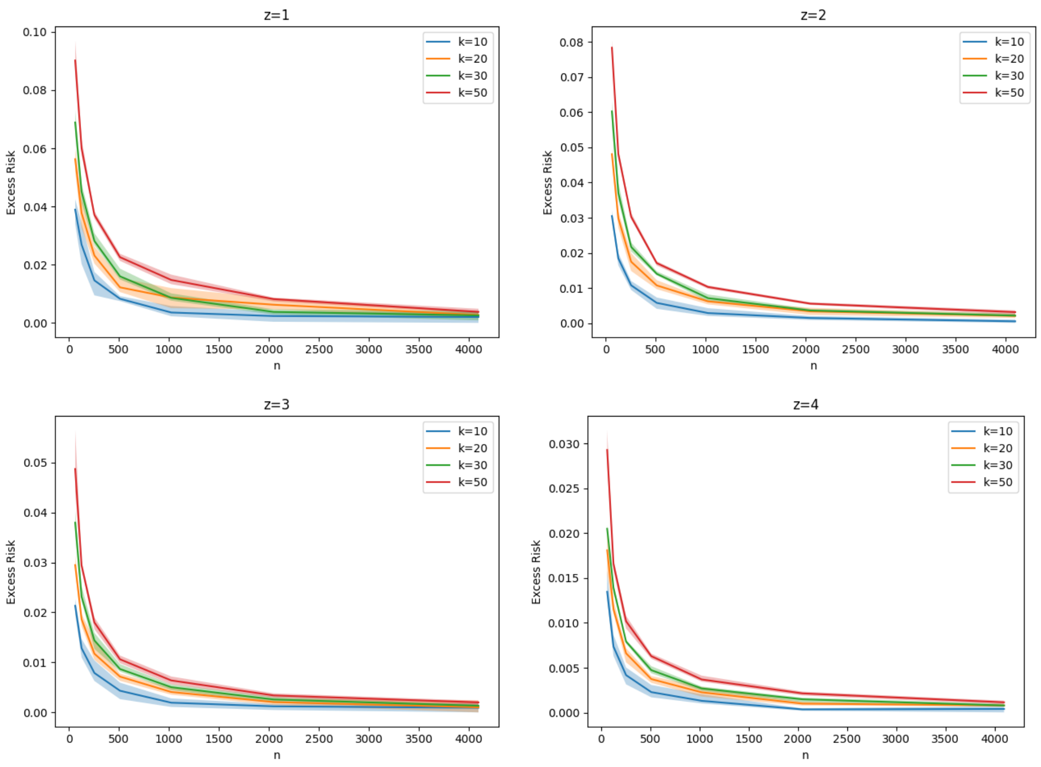


Figure 3.5. Excess risk for center-based clustering on the MNIST dataset. The shaded areas indicate the maximal and minimal deviation for the respective sample sizes.

Table 3.4. Best fit line for subspace clustering on Covtype and Mushroom (left to right)

j	z	c	q_1	q_2
1	1	0.1	0.45	0.54
1	2	$2 \cdot 10^{-2}$	0.48	0.51
1	3	$3 \cdot 10^{-4}$	0.46	0.53
1	4	$4 \cdot 10^{-5}$	0.46	0.52
2	1	$8 \cdot 10^{-2}$	0.48	0.51
2	2	$2 \cdot 10^{-3}$	0.47	0.51
2	3	$4 \cdot 10^{-5}$	0.46	0.53
2	4	$2 \cdot 10^{-6}$	0.46	0.52
5	1	$8 \cdot 10^{-3}$	0.48	0.51
5	2	$5 \cdot 10^{-5}$	0.46	0.53
5	3	$4 \cdot 10^{-7}$	0.47	0.52
5	4	$3 \cdot 10^{-9}$	0.47	0.51

j	z	c	q_1	q_2
1	1	$1 \cdot 10^{-1}$	0.48	0.51
1	2	$1 \cdot 10^{-1}$	0.48	0.51
1	5	$1 \cdot 10^{-1}$	0.49	0.49
2	1	$7 \cdot 10^{-2}$	0.48	0.51
2	2	$6 \cdot 10^{-2}$	0.50	0.49
2	5	$6 \cdot 10^{-2}$	0.49	0.48
3	1	$4 \cdot 10^{-2}$	0.49	0.50
3	2	$3 \cdot 10^{-2}$	0.49	0.50
3	5	$3 \cdot 10^{-2}$	0.49	0.49
4	1	$2 \cdot 10^{-2}$	0.49	0.50
4	2	$2 \cdot 10^{-2}$	0.49	0.50
4	5	$1 \cdot 10^{-2}$	0.48	0.50

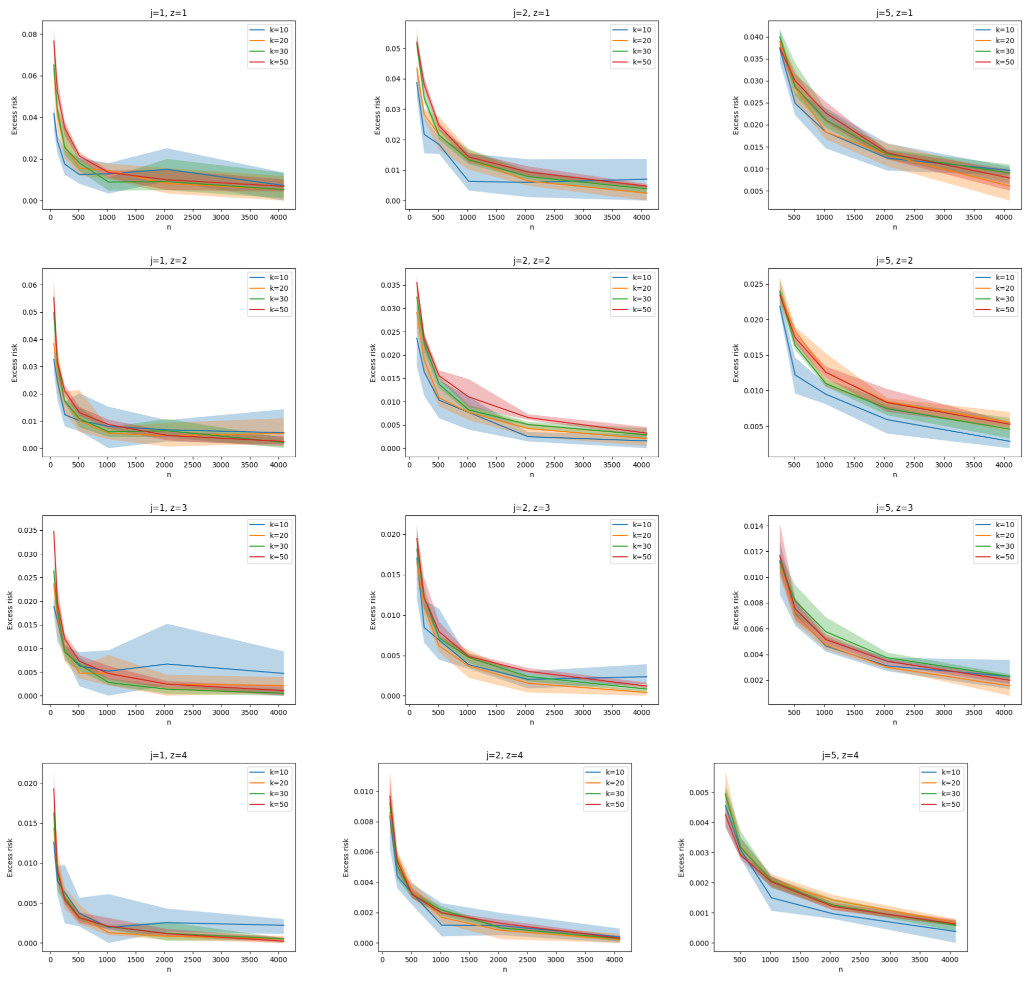


Figure 3.6. Excess risk for subspace clustering on the Mushroom dataset. The shaded areas indicate min/max values

Table 3.5. Best fit line for subspace clustering on Skin-Nonskin

j	z	c	q_1	q_2
1	1	$1 \cdot 10^{-2}$	0.48	0.50
1	2	$3 \cdot 10^{-3}$	0.45	0.53
2	1	$2 \cdot 10^{-3}$	0.46	0.53
2	2	$2 \cdot 10^{-4}$	0.46	0.53
3	1	$4 \cdot 10^{-4}$	0.46	0.53
3	2	$2 \cdot 10^{-5}$	0.46	0.53
4	1	$9 \cdot 10^{-5}$	0.46	0.53
4	2	$3 \cdot 10^{-6}$	0.46	0.53

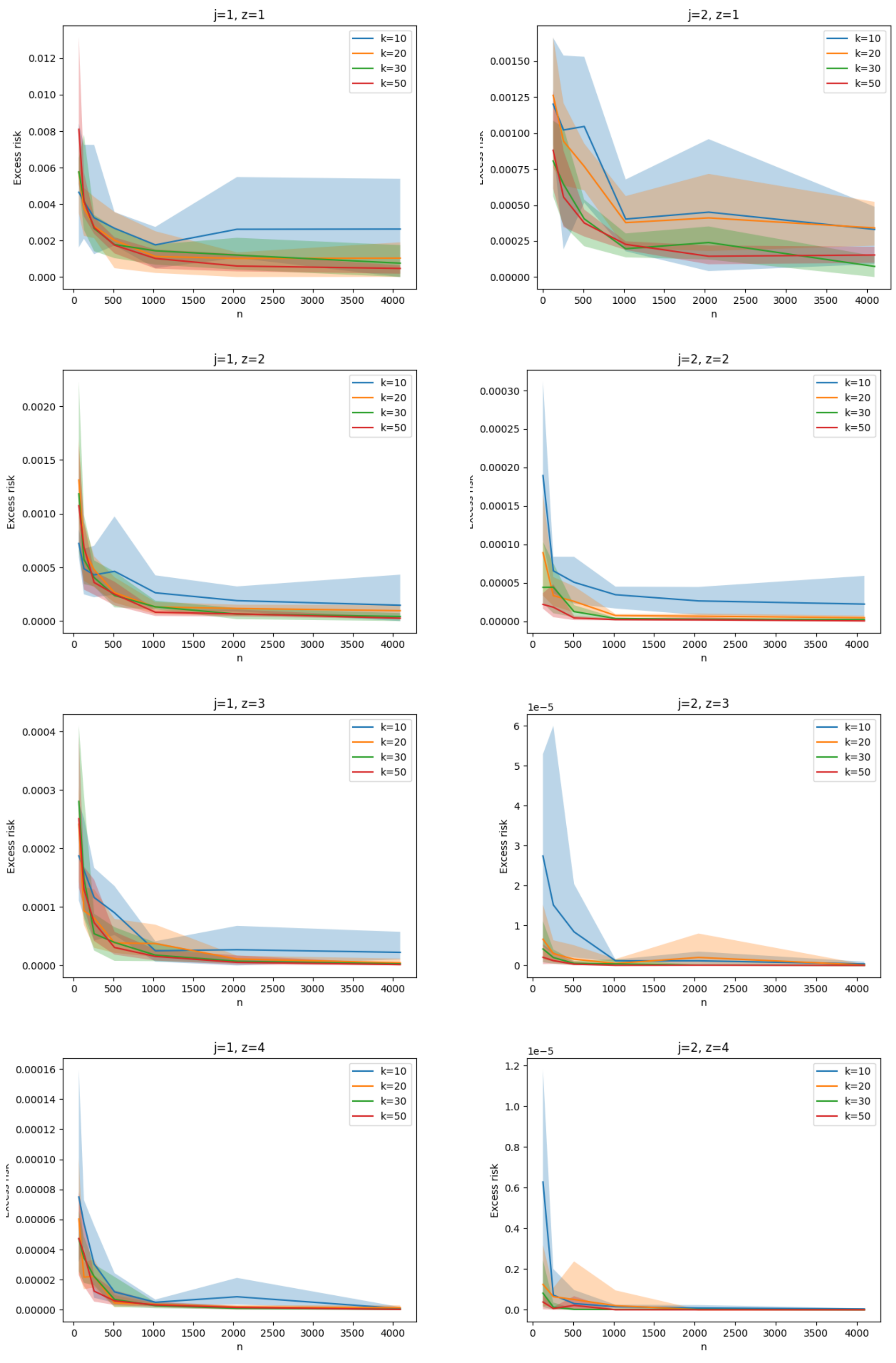


Figure 3.7. Excess risk for subspace clustering on the Skin_Nonskin dataset. The shaded areas indicate min/max values

Chapter 4

Newron: A New Generalization of the Artificial Neuron to Enhance the Interpretability of Neural Networks while Preserving Expressive Capabilities

Neural Networks (NNs) have now become the *de facto* standard in most Artificial Intelligence (AI) applications. The world of Machine Learning has moved towards Deep Learning, i.e., a class of NN models that exploit the use of multiple layers in the network to obtain the highest performance.

Research in this field has focused on methods to increase the performance of NNs, in particular on which activation functions (Apicella et al., 2021) or optimization method (Sun et al., 2019) would be best. Higher performances come at a price: Arrieta et al. (2020) show that there is a trade-off between interpretability and accuracy of models. Explainable Artificial Intelligence (XAI) is a rapidly growing research area producing methods to interpret the output of AI models in order to improve their robustness and safety (see e.g. Ghorbani et al. (2019) and Bhatt et al. (2019)). Deep Neural Networks (DNNs) offer the highest performance at the price of the lowest possible interpretability. It is an open challenge to attain such high performance without giving up on model interpretability.

The simplest solution would be to use a less complex model that is natively interpretable, e.g., decision trees or linear models, but those models are usually less effective than NNs. We ask the following question:

RQ5 *Is it possible to design a novel neural network structure that makes the whole model interpretable without sacrificing effectiveness and expressiveness?*

NNs are black-box models: we can only observe their input and output values with no clear understanding of how those two values are correlated according to the model's parameters. Although a single neuron in the NN performs a relatively simple linear combination of the inputs, there is no clear and straightforward link between the parameters estimated during the training and the functioning of the network, mainly because of the stacking of multiple layers and non-linearities.

In this work, we propose a generalization of the standard neuron used in neural networks that can also represent new configurations of the artificial neuron. Thus, we discuss a specific example that allows us to interpret the functioning of the network itself. Like the standard neuron, ours can also be used to stack multiple layers in sequence, i.e. to generate DNNs.

We focus our efforts on tabular data since we investigate how NEWRON works only in the case of fully connected NNs. It is more straightforward to produce human-readable rules for this kind of data. We also remark that our goal is not to improve the performance of NNs, but rather to create interpretable versions of NNs that perform as well as other interpretable models (e.g., linear/logistic regression, decision trees, etc.) and similarly to standard NNs, when trained on the same data.

4.0.1 Motivating example

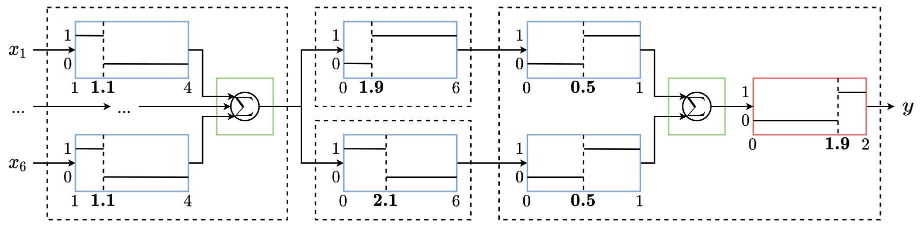


Figure 4.1. An example of a network for the MONK-2 dataset. x_i are the inputs, y is the output. The red and blue rectangles represent the plot of functions, with input range on the x-axis and output on the y-axis. The green rectangles contain the aggregation function. The numbers in bold represent the thresholds for the step functions.

Consider a simple dataset: MONK's[1]. Each sample consists of 6 attributes, which take integer values between 1 and 4 and a class label determined by a decision rule based on the 6 attributes. For example, in MONK-2, the rule that defines the class for each sample is the following: "*exactly two*" out of the six attributes are equal to 1.

It is impossible to intuitively recover rules from the parameter setting from a traditional, fully connected NN.

We shall see in the following that our main idea is that of inverting the activation and aggregation. In NEWRON the nonlinearity directly operates on the input of the neuron. The nonlinearity acts as a thresholding function to the input, making it directly interpretable as a (fuzzy) logical rule by inspecting its parameters. Consider the following network, represented in Figure 4.1: 2 hidden layers, the first with 1 neuron, the second with 2 neurons, and 1 output neuron. The x_i's are the inputs of the model, y is the output.

We present the form of a typical architecture composed by NEWRON in Figure 4.1. We show how we can interpret the parameters obtained from a trained network. The rectangles represent the plot of a function that divides the input domain into two intervals, separated by the number below the rectangle, taking values 1 and 0.

The functions that process the input give output 1 only if the input is less than 1.1, given that inputs are integers and assume values only in $\{1, 2, 3, 4\}$, this means "if $x_i = 1$". The sum of the output of all these functions, depicted in the green rectangle, then represents the degree of soundness of those rules are.

[1]https://archive.ics.uci.edu/ml/datasets/MONK%27s+Problems

The second layer has two neurons: the first outputs 1 if it receives an input greater than 1.9, i.e. if at least 2 of the rules $x_i = 1$ are valid, while the second outputs 1 if it receives an input less than 2.1, i.e. if 2 or less of the rules $x_i = 1$ are valid. Notice that the two neurons are activated simultaneously only if $x_i = 1$ is true for exactly two attributes.

In the last layer, functions in the blue rectangles receive values in $\{0, 1\}$ and do not operate any transformation, keeping the activation rules unchanged. The sum of the outputs of these functions is then passed to the function in the red rectangle. This function outputs 1 only if the input is greater than 1.9. Since the sum is limited in $0, 1, 2$, this happens only when it receives 2 as input, which occurs only if the two central neurons are activated. As we have seen, this only applies if exactly 2 of the rules $x_i = 1$ are valid.

So we can conclude that the network gives output 1 just if "*exactly two*" of $\{x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1, x_5 = 1, x_6 = 1\}$ are true.

4.0.2 Contributions

The main contributions of this work are the following:

- We propose Newron, a generalization of the McCulloch-Pitt neuron allowing the definition of new artificial neurons. We show how special cases of Newron may pave the way towards interpretable, white-box neural networks.

- We prove the universal approximation theorem for three specializations of Newron, demonstrating that the new model does not lose any representation power in those cases.

- We experiment on several tabular datasets showing that Newron allows learning accurate Deep Neural models, beating interpretable by design models such as Decision Trees and Logistic Regression.

4.1 Related work

Rosenblatt (1958) introduced the single artificial neuron: the Perceptron. The Perceptron resembles the functioning of the human/biological neuron, where the signal passing through the neuron depends on the intensity of the received signal, the strength of the synapses, and the receiving neuron's threshold. In the same way, the Perceptron makes a linear combination of the inputs received and is only activated if the result exceeds a certain threshold. Over the years, various improvements to neural networks have been proposed: Recurrent Units, Convolutional Layers, and Graph Neural Networks, but for Fully Connected NNs, research efforts have mainly focused on finding more efficient activation functions (Apicella et al., 2021). Two works that have focused on modifying the internal structure of the neuron are those of Kulkarni and Venayagamoorthy (2009), and Fan et al. (2018). In the former, a neuron is introduced that performs both a sum and a product of the inputs in parallel, applies a possibly different activation function for the two results, and then sums the two outcomes. Despite promising results, given the use of fewer parameters, better performance, and reduced training time compared to standard MLPs and RNNs, the proposed neuron, rather than being a generalization, is a kind of union between two standard neurons, one of which uses the product, instead of sum, as aggregation function. In the second paper, starting from the notion that the traditional neuron performs a first-order Taylor approximation, the authors propose a neuron using a second-order Taylor approximation. Although this improves the capacity of a single neuron, the authors do not demonstrate any gains in terms of training time or convergence. Indeed, this can be considered a particular case of the higher-order neural units (HONUs) (see, e.g., Gupta et al. (2013)), i.e., a type of neurons that, by increasing the degree of the polynomial computed within them, try to capture the higher-order correlation between the input patterns. Recent works that focus on interpretation at neuron level (Dalvi et al., 2019a,b; Heo et al., 2019; Nam et al., 2020) often concentrate on extracting the most relevant neurons for a given task, but mostly deal with Recurrent or Convolutional neural networks. Although not designing an alternative version of the neuron, Yang et al. (2018) proposes an alternative neural network structure, based on a Binning Layer, which divides the single input features into several bins, and a Kronecker Product Layer, which takes into account all the possible combinations between bins. The parameters estimated during training can be interpreted to translate the network into a decision tree through a clever design of the equations defining the network. Although interpretable, the main issue in this work is its scalability. The Kronecker Product Layer has an exponential complexity that makes training time unfeasible when the number of features grows.

4.2 The NEWRON structure

A neuron, in the classical and more general case, is represented by the equation $y = f\left(b + \sum_{i=1}^{n} w_i x_i\right)$.

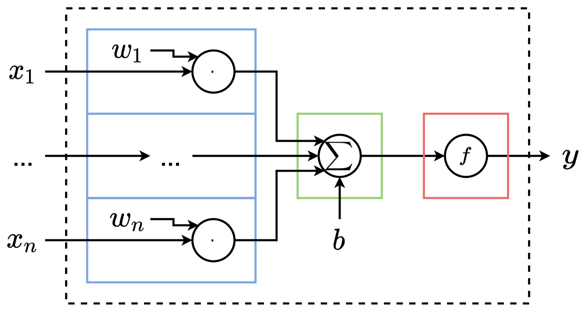


Figure 4.2. Structure of the standard artificial neuron. w_i and b are respectively weights and bias. f is the activation function. x_i's are the inputs and y is the output.

b is called the bias, w_i are the weights, and x_is are the inputs. f represents the activation function of the neuron. Usually, we use the sigmoid, hyperbolic tangent, or ReLU functions.

We first generalize the above equation, introducing NEWRON as follows:

$$y = f\left(G_{i=1}^{n}\left(h_i(x_i)\right)\right) \tag{4.1}$$

Each input is first passed through a function h_i, which we will call *processing function*, where the dependence on i indicates different parameters for each input. G, instead, represents a generic aggregation function.

Using NEWRON notation, the standard artificial neuron would consist of the following: $h_i(x_i) = w_i x_i$, $G = \sum$, and $f(z) = f^*(z + b)$.

G does not have any parameters, while b parametrizes the activation function.

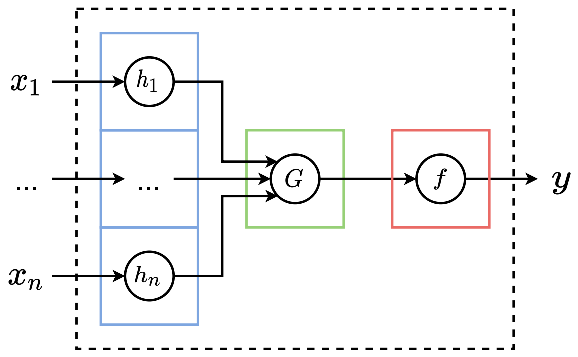


Figure 4.3. Structure of NEWRON, the generalized artificial neuron. The blue rectangles represent the processing function sections, the green rectangles contain the aggregation function, and the red rectangles represent the activation part. Same colors are also used in Figure 4.2

4.2.1 Inverted Artificial Neuron (IAN)

We present 3 novel structures characterized by an inversion of the aggregation and activation functions. We name this architectural pattern: Inverted Artificial Neuron (IAN). In all the cases we consider the sum as the aggregation function and do not use any activation function: $G = \sum$, and $f(z) = z$.

Heaviside IAN

The first case we consider uses a unit step function as activation. This function, also called the Heaviside function, is expressed by the following equation:

$$H(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{4.2}$$

According to (4.1) we can define the processing function as follows:

$$h_i(x_i) = H(w_i(x_i - b_i)) = \begin{cases} H(w_i) & x_i \geq b_i \\ 1 - H(w_i) & x_i < b_i \end{cases} \tag{4.3}$$

where w_i and b_i are trainable parameters.

Sigmoid IAN

We cannot train the Heaviside function using gradient descent, and it represents a decision rule that in some cases is too restrictive and not “fuzzy” enough to deal with constraints that are not clear-cut.

A natural evolution of the unit step function is therefore the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. This function ranges in the interval $(0, 1)$, is constrained by a pair of horizontal asymptotes, is monotonic and has exactly one inflection point.

The sigmoid function can be used as a processing function with the following parameters: $h_i(x_i) = \sigma(w_i(x_i - b_i))$.

Product of hyperbolic tangents IAN

Another option we consider as a processing function is the multiplication of hyperbolic tangent (tanh). For simplicity, we will use the term “tanh-prod”.

The tanh function $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$ is on its own very similar to the sigmoid. An interesting architecture is that using M tanh simultaneously. Each tanh applies its own weights, on each individual input.

While the sigmoid is monotonic with only one inflection point, roughly dividing the input space into two sections, the multiplication of tanh, by being not monotonic, allows us to divide the input space into several intervals. The multiplication would remain in $(-1, 1)$, but can be easily rescaled to $(0, 1)$.

We can therefore write the processing function in the case of the tanh multiplication as follows:

$$h_i(x_i) = \frac{\left(\prod_{m=1}^{M} \tanh(w_{im}(x_i - b_{im}))\right) + 1}{2} \tag{4.4}$$

Note how, in this case, the weights depend on both the input i and the m-th function. Such a neuron will therefore have M times more parameters than the Heaviside and sigmoid cases.

Output layer

The output layer would produce values ranging in the interval $(0, N)$ ($\{0, 1, ..., N\}$ for the Heaviside case), where N represents the number of neurons in the penultimate layer. This is because the last neuron makes the sum of N processing functions restricted in the interval $(0, 1)$ ($\{0, 1\}$ for the Heaviside case). To allow the last layer to have a wider output range and thus make our network able to reproduce a wider range of functions, we modify the last layer processing function h_i^* as follows: $h_i^*(x_i) = \alpha_i h_i(x_i)$,

where α_i are trainable parameters.

In the same way, as for a traditional neural network, it is important, in the output layer, to choose an adequate activation function. We need, indeed, to match the range of the output of the network and the range of the target variable. In particular, in the case of output in $(0, 1)$, we use a sigmoid centered in b^*: $f^*(z) = \sigma(z - b^*)$

In the case of a classification problem with more than 2 classes, a softmax function ($s(z_j) = \frac{e^{z_j}}{\sum_l e^{z_l}}$) is used to output probabilities.

Note(s)

The writing $w(x - b)$ is theoretically identical to that $w^*x + b^*$, where simply $w^* = w$ and $b^* = -bw$. This notation allows us to interpret the weights directly. From b, we already know the inflection point of the sigmoid; while looking at w, we immediately understand its direction.

4.3 Interpretability

Arrieta et al. (2020) presented a well-structured overview of concepts and definitions in the context of Explainable Artificial Intelligence (XAI).

They make a distinction among the various terms that are mistakenly used as synonyms for interpretability. According to them:

- **Interpretability:** is seen as a passive feature of the model and represents the ability of a human to understand the underlying functioning of a decision model, focusing more on the cause-effect relationship between input and output.

- **Transparency:** very similar to interpretability, as it represents the ability of a model to have a certain degree of interpretability. There are three categories of transparency, representing the domains in which a model is interpretable. Simulatable models can be emulated even by a human. Decomposable models must be explainable in their individual parts. For algorithmically transparent models, the user can understand the entire process followed by an algorithm to generate the model parameters and how the model produces an output from the input.

- **Explainability:** can be seen as an active feature of a model, encompassing all actions that can detail the inner workings of a model. The explanation represents a kind of interface between a human and the model and must at the same time represent well the functioning of the model and be understandable by humans.

In this chapter, we show decomposable models that, in some cases, are also algorithmically transparent.

4.3.1 Heaviside

The interpretability of an architecture composed of Heaviside IANs has to be analyzed by discussing its four main sections separately.

First layer - Processing function

A single processing function $h(x) = H(w(x-b))$ divides the space of each variable x in two half-lines starting from b, one of which has a value of 1 and one of which has a value of 0, depending on the sign of w.

Aggregation

Using sum as the aggregation function, the output takes values in $\{0, 1, ..., n\}$; where 0 corresponds to a deactivation for each input, and n represents an activation for all inputs, and the intermediate integer values $\{1, 2, ...k, ..., n-1\}$ represent activation for k of inputs.

$$y = \sum_{i=1}^{n} h_i^* = \begin{cases} n & h_i^* = 1 \ \forall i \in \{1, ..., n\} \\ k & h_i^* = 1 \ i \in S \subseteq \{1, ..., n\}, |S| = k \\ 0 & h_i^* = 0 \ \forall i \in \{1, ..., n\} \end{cases} \tag{4.5}$$

where we simplified the notation using $h_i^* = h_i(x_i)$.

2+ Layer - Processing function

Let us define an M-of-N rule as true if at least M of the N rules of a given set are true.

The Heavisides of the layers after the first one receive values in $\{0, 1, ..., n\}$, where n represents the number of inputs of the previous layer. In the case where $0 \leq b \leq n$ and $w > 0$, the Heaviside will output 1 only if the input received is greater than or equal to b, therefore only if at least $\lceil b \rceil$ of the rules R_i of the previous layer are true, which corresponds to a rule of the type $\lceil b \rceil - of - \{R_1, R_2, ..., R_n\}$. In the opposite case, where $0 \leq b \leq n$ and $w < 0$, Heaviside will output 1 only if the input received is less than or equal to b, so only if no more than $\lfloor b \rfloor$ of the rules of the previous layer are true. This too can be translated to an M-of-N rule, inverting all rules R_j and setting M as $\lceil n - b_i \rceil$: $\lceil n - b_i \rceil - of - \{\neg R_1, \neg R_2, ..., \neg R_n\}$.

Last layer - Aggregation

In the last layer we have to account for the α factors used to weigh the contribution of each input:

$$y = \sum_{i=1}^{n} \alpha_i h_i(x_i) = \sum_{i=1}^{n} \alpha_i H(w_i(x_i - b_i)) \tag{4.6}$$

We have an activation rule for each of the n Heavisides forcing us to calculate all the 2^n possible cases. The contribution of each input is exactly α_i. So, the output corresponds to the sum of the α_i's for each subset of inputs considered.

4.3.2 Sigmoid

In the case of sigmoid IAN, b_i represents the inflection point of the function, while the sign of w_i tells us in which direction the sigmoid is oriented; if positive, it is monotonically increasing from 0 to 1, while if negative, it is monotonically decreasing from 1 to 0. The value of w_i indicates how fast it transitions from 0 to 1, and if it tends to infinity, the sigmoid tends to the unit step function.

Sigmoid interpretation

The sigmoid can be interpreted as a fuzzy rule of the type $x_i > b_i$ if $w_i > 0$ or $x_i < b_i$ if $w_i < 0$, where the absolute value of w_i indicates how sharp the rule is. The case $w_i = 0$ will always give value 0.5, so that the input does not have any influence on the output.

If w_i is very large, the sigmoid tends to the unit step function. If, on the other hand, w_i takes values for which the sigmoid in the domain of x_i resembles a linear function, what we can say is that there is a direct linear relationship (or inverse if $w_i < 0$) with the input.

The fuzzy rule can be approximated by its stricter version $x_i > b_i$, interpreting fall under the methodology seen for Heaviside. However, this would result in an approximation of the operation of the network.

It is more challenging to devise clear decision rules when we add more layers. Imagine, as an example, a second layer with this processing function:

$$h(y) = \sigma(w^*(y - b^*)) \tag{4.7}$$

where y is the aggregation performed in the previous layer of the outputs of its processing functions, its value roughly indicates how many of the inputs are active. In the second layer, consider as an example a value of $w^* > 0$. To have an activation, this means that we might need k inputs greater than or equal to b^*/k. Although this does not deterministically indicate how many inputs we need to be true, we know how the output changes when one of the inputs changes.

The last case to consider takes into account the maximum and minimum values that the sigmoid assumes in the domain of x. If they are close to each other, that happens when w is very small, the function is close to a constant bearing no connection with the input.

4.3.3 Product of hyperbolic tangents

The multiplication of tanh has more expressive power, being able to represent both what is represented with the sigmoid, as well as intervals and quadratic relations.

tanh-prod Interpretation

In this case, it is not possible to devise as quickly as in the previous case decision rules. Indeed, it is still possible to observe the trend of the function and draw some conclusions. When the product of the two tanh resembles a sigmoid, we can follow the interpretation of the sigmoid case. In other cases, areas with quadratic relations can occur, i.e., bells whose peak indicates a more robust activation or deactivation for specific values.

4.3.4 Summary of Interpretation

The advantage of this method lies in the fact that it is possible to analyze each input separately in each neuron, thus easily graph each processing function. Then, based on the shape taken by the processing function, we can understand how the input affects the output of a neuron.

The Heaviside is the most interpretable of our models, allowing a direct generation of decision rules.

Sigmoid and tanh-prod cases depend on the parameter w. When it is close to 0, the activation is constant regardless of the input. When w is large enough, the processing function is approximately a piecewise constant function taking only values 0 and 1.

In all the other cases, the processing function approximates a linear or bell-shaped function. Even if we can not derive exact decision rules directly from the model, in these cases, we can infer a linear or quadratic relation between input and output.

Each layer aggregates the interpretations of the previous layers. For example, the processing function of a second layer neuron gives a precise activation when its input is greater than a certain threshold, i.e., the bias b of the processing function. The output of the neuron of the first layer must exceed this threshold, and this happens if its processing functions give in output values whose sum exceeds this threshold.

A separate case is the last layer, where the α parameters weigh each of the interpretations generated up to the last layer.

We can interpret a traditional individual neuron as a linear regressor. However, when we add more layers, they cannot be interpreted. Our structure, instead, remains interpretable even as the number of layers increases.

4.4 Universality

A fundamental property of neural networks is that of universal approximation. Under certain conditions, multilayer feed-forward neural networks can approximate any function in a given function space. In Cybenko (1989) it is proved that a neural network with a hidden layer and using a continuous sigmoidal activation function is dense in $C(I_n)$, i.e., the space of continuous functions in the unit hypercube in $\mathbb{R}^n$. (Hornik et al., 1989) generalized to the larger class of all sigmoidal functions.

To make the statement of theorems clearer we recall that the structure of a

	IAN models			Interpretable models		Non-interpretable models	
Dataset	Heaviside	sigmoid	tanh-prod	LR	DT	GBDT	NN
adult	80.2 (±0.06)	**82.6 (±0.05)**	82.3 (±0.06)	76.2 (±0.07)	81.5 (±0.06)	87.5 (±0.05)	83.1 (±0.06)
australian	86.5 (±0.51)	87.0 (±0.5)	**88.7 (±0.4)**	**88.7 (±0.4)**	87.0 (±0.41)	90.2 (±0.47)	88.0 (±0.4)
b-c-w	**98.9 (±0.16)**	**98.9 (±0.16)**	**98.9 (±0.16)**	97.8 (±0.23)	97.7 (±0.23)	98.3 (±0.21)	98.9 (±0.17)
car	95.1 (±0.2)	95.9 (±0.21)	**100.0 (±0.0)**	51.4 (±0.45)	98.5 (±0.11)	100.0 (±0.0)	99.8 (±0.04)
cleveland	**65.6 (±1.02)**	60.1 (±1.1)	62.9 (±1.13)	60.8 (±1.13)	53.6 (±1.19)	61.5 (±1.01)	65.6 (±1.01)
crx	86.2 (±0.51)	85.4 (±0.58)	86.5 (±0.5)	84.6 (±0.45)	**88.0 (±0.42)**	82.9 (±0.58)	87.7 (±0.44)
diabetes	73.3 (±0.56)	72.7 (±0.68)	**76.1 (±0.61)**	75.6 (±0.6)	74.1 (±0.63)	75.1 (±0.64)	74.2 (±0.65)
german	**78.2 (±0.53)**	77.0 (±0.53)	75.5 (±0.52)	75.1 (±0.52)	68.3 (±0.57)	76.6 (±0.55)	76.7 (±0.54)
glass	77.0 (±1.17)	81.6 (±1.04)	**85.6 (±1.02)**	72.1 (±1.08)	72.7 (±1.19)	87.3 (±0.9)	82.5 (±0.91)
haberman	76.9 (±0.94)	76.1 (±0.92)	**77.2 (±0.88)**	73.0 (±1.05)	64.4 (±1.08)	72.5 (±1.09)	76.1 (±0.92)
heart	**88.7 (±0.67)**	86.3 (±0.85)	82.7 (±0.8)	82.4 (±0.95)	81.4 (±1.02)	81.7 (±0.98)	82.9 (±0.95)
hepatitis	84.7 (±1.26)	**85.1 (±1.23)**	82.5 (±1.16)	79.1 (±1.45)	79.1 (±1.33)	81.7 (±1.32)	82.4 (±1.13)
image	93.0 (±0.11)	94.0 (±0.1)	**94.4 (±0.09)**	90.4 (±0.12)	90.6 (±0.12)	95.8 (±0.08)	92.6 (±0.11)
ionosphere	94.4 (±0.48)	**96.7 (±0.34)**	96.5 (±0.37)	92.0 (±0.51)	94.5 (±0.45)	95.4 (±0.37)	96.7 (±0.34)
iris	**100.0 (±0.0)**	**100.0 (±0.0)**	**100.0 (±0.0)**	**100.0 (±0.0)**	97.3 (±0.52)	97.3 (±0.52)	100.0 (±0.0)
monks-1	94.4 (±0.21)	**100.0 (±0.0)**	**100.0 (±0.0)**	66.0 (±0.46)	90.6 (±0.27)	100.0 (±0.0)	100.0 (±0.0)
monks-2	**100.0 (±0.0)**	**100.0 (±0.0)**	**100.0 (±0.0)**	54.5 (±0.45)	82.7 (±0.33)	94.2 (±0.21)	87.6 (±0.27)
monks-3	97.1 (±0.15)	97.1 (±0.15)	97.1 (±0.15)	81.2 (±0.31)	**97.2 (±0.16)**	96.2 (±0.16)	90.3 (±0.25)
sonar	93.3 (±0.74)	**96.8 (±0.48)**	95.2 (±0.53)	89.5 (±0.75)	83.4 (±0.98)	88.1 (±0.9)	89.4 (±0.87)
bisector	98.9 (±0.13)	99.3 (±0.09)	99.3 (±0.09)	**100.0 (±0.0)**	97.7 (±0.18)	98.3 (±0.16)	100.0 (±0.0)
xor	**100.0 (±0.0)**	**100.0 (±0.0)**	99.2 (±0.11)	53.2 (±0.65)	99.2 (±0.12)	100.0 (±0.0)	100.0 (±0.0)
parabola	98.8 (±0.15)	**100.0 (±0.0)**	99.6 (±0.07)	77.8 (±0.52)	97.6 (±0.18)	97.7 (±0.17)	100.0 (±0.0)
circle	96.8 (±0.22)	99.3 (±0.1)	**99.6 (±0.07)**	52.4 (±0.67)	98.8 (±0.13)	97.6 (±0.2)	99.2 (±0.11)

Table 4.1. Datasets accuracy ($\pm$ 95^{th} percentile standard error) results of the best performing model. In **bold** we indicate the best performing model amongst the interpretable ones. If GBDT or NN exceeds this accuracy, the corresponding result is underlined.

two-layer network with IAN neurons and a generic processing function h is

$$\psi(x) = \sum_{j=1}^{N} \alpha_j h_j (\sum_{i=1}^{n} h_{ij}(x_i)) \tag{4.8}$$

where $\alpha_j \in \mathbb{R}\ \forall j \in \{1, ..., N\}$.

When the processing function is the Heaviside function we proved that the network can approximate any continuous function on unit hypercube, I_n, Lebesgue measurable functions on I_n and functions in $L^p(A, \mu)$ for $1 \le p < \infty$, with μ being a Radon measure and $A \in \mathcal{B}(\mathbb{R}^n)$ a Borel set. More precisely, the following theorems hold; we first state all the theorems and later prove each of them.

Theorem 4.4.1. *When the processing function is the Heaviside function the finite sums of the form (4.8) are dense in $L^p(A, \mu)$ for $1 \le p < \infty$, for any $A \in \mathcal{B}(\mathbb{R}^n)$ - $\mathcal{B}$ denote the Borel σ–algebra - and μ Radon measure on $(A, \mathcal{B}(A))$.*

Theorem 4.4.2. *When the processing function is the Heaviside function, the finite sums of the form (4.8) are dense in the space of Lebesgue measurable functions on I_n w.r.t the convergence in measure.*

Theorem 4.4.3. *Given $g \in C(I_n)$ and given $\epsilon > 0$ there is a sum $\psi(x)$ of the form (4.8) with Heaviside as processing function such that*

$$|\psi(x) - g(x)| < \epsilon \quad \forall x \in I_n.$$

When the processing function is the sigmoid function or tanh-prod, we proved that the finite sums of the form (4.8) are dense in $C(I_n)$.

Theorem 4.4.4. *When the processing function is a continuous sigmoidal function the finite sums of the form (4.8) are dense in $C(I_n)$.*

Theorem 4.4.5. *Let $\psi(x)$ be the family of networks defined by the equation (4.8) when the processing function is given by (4.4). This family of functions is dense in $C(I_n)$.*

Universality Theorems for Heaviside IAN

Theorem 4.4.6. *The finite sums of the form*

$$\psi(x) = \sum_{j=1}^{N} \alpha_j H(w_j \sum_{i=1}^{n} H(w_{ij}(x_i - b_{ij})) - b_j) \tag{4.9}$$

with $N \in \mathbb{N}$ and $w_{ij}, w_j, \alpha_j, b_{ij}, b_j \in \mathbb{R}$ are dense in $L^p(A, \mu)$ for $1 \leq p < \infty$, for any $A \in \mathcal{B}(\mathbb{R}^n)$ ($\mathcal{B}$ denote the Borel σ–algebra) and μ a Radon measure on $(A, \mathcal{B}(A))$.

In other words given, $g \in L^p(A, \mu)$ and $\epsilon > 0$ there is a sum $\psi(x)$ of the above form for which

$$||\psi - g||_p^p = \int_{\mathbb{R}^n} |\psi(x) - g(x)|^p d\mu(x) < \epsilon.$$

To prove that a neural network defined as in equation (4.9) is a universal approximator in L^p, for $1 \leq p < \infty$ we exploit that step functions are dense in L^p and that our network can generate step functions.

Proposition 4.4.6.1. *Let $\mathcal{R}$ be the set of the rectangles in $\mathbb{R}^n$ of the form*

$$R = \prod_{k=1}^{n} [a_k, b_k) \quad a_k, b_k \in \mathbb{R},\ a_k < b_k$$

We denote by $\mathcal{F}$ the vector space on $\mathbb{R}$ generated by $\mathbb{1}_R$, $R \in \mathcal{R}$ i.e.

$$\mathcal{F} = \Big\{ \sum_{i=1}^{m} \alpha_i \mathbb{1}_{R_i} \Big| \, m \in \mathbb{N}, \alpha_i \in \mathbb{R}, R_i \in \mathcal{R} \Big\} \tag{4.10}$$

If $A \in \mathcal{B}(\mathbb{R}^n)$, then the set

$$\mathcal{F}|_A = \Big\{ \sum_{i=1}^{m} \alpha_i \mathbb{1}_{R_i \cap A} \Big| \, m \in \mathbb{N}, \alpha_i \in \mathbb{R}, R_i \in \mathcal{R} \Big\} \tag{4.11}$$

$\mathcal{F}|_A$ is dense in $L^p(A, \mu)$ for $1 \leq p < \infty$, with μ a Radon measure on $(A, \mathcal{B}(A))$.

Proof. See chapter 3, L^p Spaces , in Cannarsa and D'Aprile (2015). □

Lemma 4.4.7. *Given $\rho(x) \in \mathcal{F}$, with $\mathcal{F}$ defined as in equation (4.10), there exists a finite sum $\psi(x)$ of the form (4.9) such that $\rho(x) = \psi(x)\ \forall x \in \mathbb{R}^n$.*

Proof. To prove that a neural network described as in equation (4.9) can generate step functions we proceed in two steps. First, we show how we can obtain the indicator functions of orthants from the first layer of the network. Then we show how, starting from these, we can obtain the step functions.

An *orthant* is the analogue in n-dimensional Euclidean space of a quadrant in $\mathbb{R}^2$ or an octant in $\mathbb{R}^3$. We denote by *translated orthant* an orthant with origin in a point different from the origin of the Euclidean space O. Let A be a point in the n-dimensional Euclidean space, and let us consider the intersection of n mutually orthogonal half-spaces intersecting in A. By independent selections of half-space signs with respect to A (i.e. to the right or left of A) 2^n orthants are formed.

Now we shall see how to obtain translated orthant with origin in in a point A of coordinates $(a_1, a_2, ..., a_n)$ from the first layer of the network i.e. $\sum_{i=1}^{n} H(w_i(x_i - b_i))$.

For this purpose we can take $w_i = 1 \quad \forall i \in \{1, ..., n\}$.

The output of $\sum_{i=1}^{n} H(x_i - b_i) \in \{0, ..., n\}$ and depends on how many of the n Heaviside functions are activated. We obtain the translated orthant with origin in A by choosing $b_i = a_i \quad \forall i \in \{1, ..., n\}$. In fact,

$$H(x_i - a_i) = \begin{cases} 0 & \text{if } x_i < a_i \\ 1 & \text{if } x_i \geq a_i. \end{cases}$$

The i-th Heaviside is active in the half-space $x_i \geq a_i$ delimited by the hyperplane $x_i = a_i$ that is orthogonal to the i-th axis. Therefore, the Euclidian space $\mathbb{R}^n$ is divided in 2^n regions according to which value the function $\sum_{i=1}^{n} H(x_i - a_i)$ takes in each region. See Figure 4.4 for an example in $\mathbb{R}^2$.

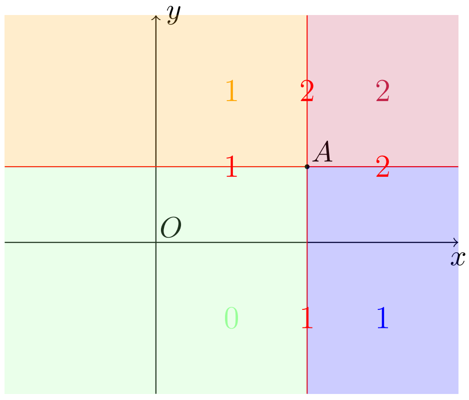


Figure 4.4. Partition of $\mathbb{R}^2$, according to output of the function $H(x_1 - a_1) + H(x_2 - a_2)$. A is a point of coordinates (a_1, a_2).

There is only one region in which the output of the sum is n, which corresponds to the orthant in which the condition $x_i \geq a_i \; \forall i = 1, ..., n$ holds. We denote it as *positive othant* (the red colored orthant in the example shown in Figure 4.4).

Going back to equation (4.9), let us now consider the Heaviside function applied after the sum. As before, we can choose $w_j = 1$. If we take $b_j > n - 1$, the value of the output is 0 for each of the 2^n orthants except for the positive orthant. This way, we get the indicator function of the positive orthant.

The indicator function of a rectangle in $\mathcal{R}$ can be obtained as a linear combination of the indicator function of the positive orthants centered in the vertices of the rectangle. See Figure 4.5 for an example of the procedure in $\mathbb{R}^2$.

In general, the procedure involves considering a linear combination of indicator functions of positive orthants centered in the vertices of the rectangle in such a way that opposite values are assigned to the orthants corresponding to adjacent vertices.

For example, suppose we want to obtain the indicator function of the right-closed left-open square $[0, 1)^2$ in $\mathbb{R}^2$ (see the illustration in Figure 4.5). Denoting by $\mathbb{1}_{(x_P, y_P)_\llcorner}$ the indicator function of the positive orthant centered in the point of coordinates (x_P, y_P), we can write:

$$\mathbb{1}_{[0,1)^2} = \mathbb{1}_{(0,0)_\llcorner} - \mathbb{1}_{(1,0)_\llcorner} - \mathbb{1}_{(0,1)_\llcorner} + \mathbb{1}_{(1,1)_\llcorner}.$$

Now suppose we want the linear combination of the indicator functions of K rectangles with coefficents $\alpha_1, ...\alpha_K$. With suitably chosen coefficients the indicator function of a rectangle can be written as

$$\sum_{l=1}^{2^n} (-1)^l H(w_{jl} \sum_{i=1}^{n} H(w_{ij}(x_i - b_{ij})) - b_{jl})$$

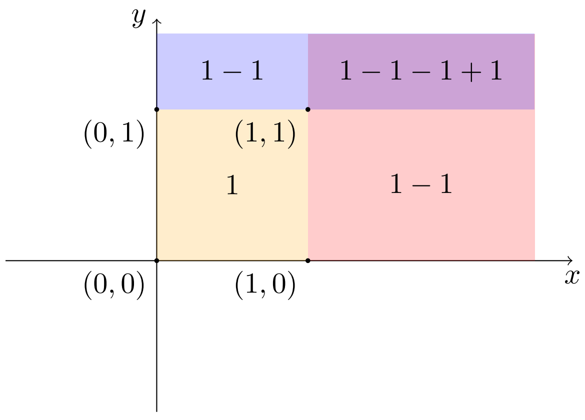


Figure 4.5. How to obtain the indicator function on the square $[0,1)^2$ from the linear combination of four indicator functions of positive orthants centered in the vertices of $[0,1)^2$. $\mathbb{1}_{[0,1)^2} = \mathbb{1}_{(0,0)_\llcorner} - \mathbb{1}_{(1,0)_\llcorner} - \mathbb{1}_{(0,1)_\llcorner} + \mathbb{1}_{(1,1)_\llcorner}$. The numbers in the orthants shows the sum of the indicator functions that are active in that orthant. For instance if $x = (x_1, x_2)$ belongs to the blue part of the plane, i.e. it is true that $0 < x_1 < 1$ and $x_2 > 1$, we have that $\mathbb{1}_{(0,0)_\llcorner}(x) - \mathbb{1}_{(1,0)_\llcorner}(x) - \mathbb{1}_{(0,1)_\llcorner}(x) + \mathbb{1}_{(1,1)_\llcorner}(x) = 1-0-1+0 = 1-1$.

that replacing $H(w_{jl} \sum_{i=1}^{n} H(w_{ij}(x_i - b_{ij})) - b_{jl})$ by H_l, to abbreviate the notation becomes

$$\sum_{l=1}^{2^n} (-1)^l H_l.$$

The linear combination of the indicator functions of K rectangles with coefficents $\alpha_1, ...\alpha_K$ can be derived as

$$\sum_{k=1}^{K} \alpha_k \sum_{l=1}^{2^n} (-1)^l H_{lk}. \tag{4.12}$$

The summation (4.12) can be written as a single sum, defining a sequence $\beta_j = (-1)^j \alpha_m$ with $m = \lceil \frac{j}{2^n} \rceil$ for $j = 1, ..., 2^n K$. Thus (4.12) becomes

$$\sum_{j=1}^{N=2^n K} \beta_j H_j$$

that is an equation of form (4.9). We have therefore shown that for every step function ρ in $\mathcal{F}$ there are $N \in \mathbb{N}$ and α_j, w_{ij}, b_{ij}, $b_j, w_j \in \mathbb{R}$ such that the sum in equation (4.9) is equal to ρ. □

Proof of Theorem 4.4.6. The theorem follows immediately from Lemma 4.4.7 and Proposition 4.4.6.1. □

Remark 3. *In Lemma 4.4.7 we proved that a network defined as in equation (4.9) can represent functions belonging to set $\mathcal{F}$ defined as in equation (4.10). Note that if the input is a set A we can obtain functions belonging to set $\mathcal{F}|_A$. For instance, suppose $x \in [0,1]^n$, we can obtain indicator functions of other kinds of sets. If we choose $w_{ij} = 1$ and $b_{ij} < 0$ $\forall i, j$ and if we choose the weights of the second layer so that they don't operate any transformation, we can obtain the indicator function of $[0,1]^n$. By a suitable choice of parameters, (4.9) may also become the indicator* *functions of any hyperplane $x_i = 0$ or $x_i = 1$ for $i \in \{1, .., n\}$. Furthermore we can obtain any rectangle of dimension $n-1$ that belongs to an hyperplane of the form $x_i = 1$ or $x_i = 0$.*

We have proven in Lemma 4.4.7 that a network formulated as in equation (4.9) can represent step functions. By this property and by Proposition 4.4.8.1 we shall show that it can approximate Lebesgue measurable functions on any finite space, for example the unit n-dimensional cube $[0,1]^n$.

We denote by I_n the closed n-dimensional cube $[0,1]^n$. We denote by M^n the set of measurable functions with respect to Lebesgue measure m, on I_n, with the metric d_m defined as follows. Let be $f, g \in M^n$,

$$d_m(f,g) = \inf\{\epsilon > 0 : m\{x : |f(x) - g(x)| > \epsilon\} < \epsilon\}$$

We remark that d_m-convergence is equivalent to convergence in measure (see Lemma 2.1 in Hornik et al. (1989)).

Theorem 4.4.8. *The finite sums of the form (4.9) with $N \in \mathbb{N}$ and $w_{ij}, w_j, \alpha_j, b_{ij}, b_j \in \mathbb{R}$ are d_m-dense in M^n. M^n is the set of Lebesgue measurable functions on I_n .*

This means that, given g measurable with respect to the Lebesgue measure m on I_n, and given an $\epsilon > 0$, there is a sum ψ of the form (4.9) such that $d_m(\psi, g) < \epsilon$.

Proposition 4.4.8.1. *Suppose f is measurable on $\mathbb{R}^n$. Then there exists a sequence of step functions $\{\rho_k\}_{k=1}^{\infty}$ that converges pointwise to $f(x)$ for almost every x.*

Proof. See Theorem 4.3 p. 32 in Stein and Shakarchi (2005). □

Proof of Theorem 4.4.8. Given any measurable function, by Proposition 4.4.8.1 there exists a sequence of step functions that converge to it pointwise. By Lemma 4.4.7 we have that equation (4.9) can generate step functions. Now $m(I_n) = 1$ and for a finite measure space pointwise convergence implies convergence in measure, this concludes the prof. □

Remark 4. *Notice that for Theorem 4.4.8 we don't need the fact that I_n, is a closed set. The proof only requires that the domain is a bounded set (so that its Lebesgue measure is finite). The compactness of I_n will be necessary for the next theorem.*

We notice furthermore that if the function we want to approximate is in L^p we can obtain the convergence in measure from Theorem 4.4.6. Indeed from Chebyshev's inequality it follows that convergence in L^p implies convergence in measure.

Theorem 4.4.9. *Given $g \in C(I_n)$ and given $\epsilon > 0$ there is a sum $\psi(x)$ of the form (4.9) such that*

$$|\psi(x) - g(x)| < \epsilon \quad \forall x \in I_n.$$

Proof. Let g be a continuous function from I_n to $\mathbb{R}$, by the compactness of I_n follows that g is also uniformly continuous (see Theorem 4.19 p. 91 in Rudin (1976)). In other words, for any $\epsilon > 0$, there exists $\delta > 0$ such that for every $x, x' \in [0,1]^n$ such that $||x - x'||_\infty < \delta$ it is true that $|g(x) - g(x')| < \epsilon$. To prove the statement of Theorem 4.4.9, let $\epsilon > 0$ be given, and let $\delta > 0$ be chosen according to the definition of uniform continuity.

As we have already seen in Lemma 4.4.7 the neural network described in (4.9) can generate step functions with support on right-open left-closed n-dimensional rectangles and on $(n-1)$-dimensional rectangles that belongs to an hyperplane of equation $x_i = 0$ or $x_i = 1$ for some $i \in \{1, ..., n\}$ as seen in Remark 3. There exists a partition of $[0,1]^n$, $(R_1, ..., R_N)$, consisting of right-open left-closed n-dimensional rectangles and of $(n-1)$-dimensional rectangles that belongs to an hyperplane of equation $x_i = 0$ or $x_i = 1$ for some $i \in \{1, ..., n\}$, such that all side lengths are no greater than δ. An example of a set of rectangles with this property is the set of right-open left-closed cubes of side length $\frac{1}{\tilde{m}}, \tilde{m} > \lceil \frac{1}{\delta} \rceil$ with the $(n-1)$-dimensional rectangles with the same side length which we need to cover all the boundary of $[0,1]^n$ not covered by the right-open left-closed rectangles.

Suppose that for all $j \in \{1, ..., N\}$ we choose $x_j \in R_j$, and we set $\alpha_j = g(x_j)$. If $x \in [0,1]^n$ there is j so that $x \in R_j$, hence x satisfies $||x - x_j||_\infty \leq \delta$, and consequentially $|g(x) - g(x_j)| \leq \epsilon$. Therefore the step function $h = \sum_{j=1}^N \alpha_j \mathbb{1}_{R_j}$ satisfies

$$\begin{aligned} &\sup_{x \in I_n} |h(x) - g(x)| = \\ &= \sup_{j \in \{1,...,N\}} \sup_{x \in R_j} |h(x) - g(x)| = \\ &= \sup_{j \in \{1,...,N\}} \sup_{x \in R_j} |\alpha_j - g(x)| \leq \epsilon \end{aligned}$$

□

Universality Theorem for Sigmoid IAN

Definition 4.4.10. A function $\sigma : \mathbb{R} \to [0,1]$ is called sigmoidal if

$$\lim_{x \to -\infty} \sigma(x) = 0, \qquad \lim_{x \to +\infty} \sigma(x) = 1$$

Theorem 4.4.11. *Let σ be a continuous sigmoidal function. Then the finite sums of the form:*

$$\psi(x) = \sum_{j=1}^N \alpha_j \sigma(w_j(\sum_{i=1}^n \sigma(w_{ij}(x_i - b_{ij})) - b_j)) \tag{4.13}$$

with $w_{ij}, \alpha_j, b_{ij}, b_j, w_j \in \mathbb{R}$ and $N \in \mathbb{N}$ are dense in $C(I_n)$.

In other words, given a $g \in C(I_n)$ and given $\epsilon > 0$ there is a sum $\psi(x)$ of the form (4.13) such that

$$|\psi(x) - g(x)| < \epsilon \quad \forall x \in I_n.$$

Proof. Since σ is a continuous function, it follows that the set U of functions of the form (4.13) with $\alpha_j, w_{ij}, b_{ij}, w_j, b_j \in \mathbb{R}$ and $N \in \mathbb{N}$ is a linear subspace of $C(I_n)$. We claim that the closure of U is all of $C(I_n)$.

Assume that U is not dense in $C(I_n)$, let S be the closure of U, $S \neq C(I_n)$. By the Hahn-Banach theorem (see p. 104 of Rudin (1987)) there is a bounded linear functional on $C(I_n)$, call it L, with the property that $L \neq 0$ but $L(S) = L(U) = 0$.

By the Riesz Representation Theorem (see p. 40 of Rudin (1987)), the bounded linear functional L, is of the form

$$L(f) = \int_{I_n} f(x) d\mu$$

for some signed regular Borel measures μ such that $\mu(K) < \infty$ for every compact set $K \subset I_n$ (i.e. μ is a Radon measure). Hence,

$$\int_{I_n} h(x)d\mu = 0, \forall h \in U. \tag{4.14}$$

We shall prove that (4.14) implies $\mu = 0$, which contradicts the hypothesis $L \neq 0$.

Using the definition of U, equation (4.14) can also be written as

$$\sum_{j=1}^{N} \alpha_j \int_{I_n} \sigma(w_j(\sum_{i=1}^{n} \sigma(w_{ij}(x_i - b_{ij})) - b_j))d\mu = 0,$$

for any choice of $\alpha_j, w_{ij}, w_j, b_{ij}, b_j \in \mathbb{R}$ and $N \in \mathbb{N}$.

Note that for any $w, x, b \in \mathbb{R}$ we have that the continuous functions

$$\sigma_\lambda(w(x-b)) = \sigma(\lambda w(x-b) + \phi)$$

converge pointwise to the unit step function as λ goes to infinity, i.e.

$$\lim_{\lambda\to\infty} \sigma_\lambda(w(x-b)) = \gamma(w(x-b))$$

with

$$\gamma(y) = \begin{cases} 1 & \text{if } y > 0 \\ \sigma(\phi) & \text{if } y = 0 \\ 0 & \text{if } y < 0 \end{cases}$$

By hypothesis is true that for all λ_1, λ_2 in $\mathbb{R}$

$$\int_{I_n} \sigma_{\lambda_2}(w_j(\sum_{i=1}^{n} \sigma_{\lambda_1}(w_{ij}(x_i - b_{ij})) - b_j))d\mu = 0.$$

It follows that for all λ_2:

$$\lim_{\lambda_1\to\infty} \int_{I_n} \sigma_{\lambda_2}(w_j(\sum_{i=1}^{n} \sigma_{\lambda_1}(w_{ij}(x_i - b_{ij})) - b_j))d\mu = 0.$$

Now applying the Dominated Convergence Theorem (see Theorem 11.32 p 321 of Rudin (1976)) and the fact that σ is continuous:

$$\int_{I_n} \lim_{\lambda_1\to\infty} \sigma_{\lambda_2}(w_j(\sum_{i=1}^{n} \sigma_{\lambda_1}(w_{ij}(x_i - b_{ij})) - b_j))d\mu =$$
$$\int_{I_n} \sigma_{\lambda_2}(w_j(\sum_{i=1}^{n} \gamma(w_{ij}(x_i - b_{ij})) - b_j))d\mu.$$

Again, by Dominated Convergence Theorem we have:

$$\lim_{\lambda_2\to\infty} \int_{I_n} \sigma_{\lambda_2}(w_j(\sum_{i=1}^{n} \gamma(w_{ij}(x_i - b_{ij})) - b_j))d\mu =$$
$$\int_{I_n} \gamma(w_j(\sum_{i=1}^{n} \gamma(w_{ij}(x_i - b_{ij})) - b_j)))d\mu.$$

Hence we have obtained that $\forall \alpha_j, w_{ij}, b_{ij}, w_j, b_j \in \mathbb{R}$ and $\forall N \in \mathbb{N}$

$$\int_{I_n} \sum_{j=1}^{N} \alpha_j \gamma(w_j(\sum_{i=1}^{n} \gamma(w_{ij}(x_i - b_{ij})) - b_j))d\mu = 0.$$

The function γ is very similar to the Heaviside function H, the only difference is that $H(0) = 1$ while $\gamma(0) = \sigma(\phi)$. Let R_i denote an open rectangle, $\partial_a R_i$ its left boundary (i.e. the boundary of a left-closed right-open rectangle) and $\partial_b R_i$ its right boundary (i.e. the boundary of a right-closed left-open rectangle). Repeating the construction seen in Lemma 4.4.7 to obtain rectangles, with the difference that here γ takes value $\sigma(\phi)$ on the boundaries, we get that

$$\sigma(\phi)\mu(\partial_a R_i) + (1 - \sigma(\phi))\mu(\partial_b R_i) + \mu(R_i) = 0$$

for every open rectangle R_i. Taking $\phi \to \infty$, implies

$$\mu(\partial_a R_i) + \mu(R_i) = 0 \quad \forall \text{ open rectangle } R_i.$$

Every open subset A of I_n, can be written as a countable union of disjoint partly open cubes (see Theorem 1.11 p.8 of Wheeden and Zygmund (2015)). Thus, from the fact that the measure is σ-additive we have that for every open subset A of I_n, $\mu(A) = 0$. Furthermore $\mu(I_n) = 0$. To obtain I_n from

$$\sum_{j=1}^{N} \alpha_j \gamma(w_j(\sum_{i=1}^{n} \gamma(w_{ij}(x_i - b_{ij})) - b_j))$$

it is sufficient to choose the parameters so that $w_{ij}(x_i - b_{ij}) > 0 \; \forall x_i \in [0, 1]$ and so that w_j, b_j maintains the condition on the input.

Hence, $\mu(A^C) = \mu(I_n) - \mu(A) = 0$. It follows that for all compact set K of I_n, $\mu(K) = 0$.

From the regularity of the measure, it follows that μ is the null measure.

□

Universality Theorem for Product of hyperbolic tangents IAN

Theorem 4.4.12. *The finite sums of the form*

$$\begin{aligned} \psi(x) &= \sum_{j=1}^{N} \frac{\alpha_j}{2} \left[\prod_{l=1}^{M_j} \tanh(w_{jl}(z_j(x) - b_{jl})) + 1 \right] \\ z_j(x) &= \sum_{i=1}^{n} \frac{1}{2} \left[\prod_{k=1}^{m_i} \tanh(w_{ijk}(x_i - b_{ijk})) + 1 \right] \end{aligned} \tag{4.15}$$

with $w_{jl}, w_{ijk}, \alpha_j, b_{jl}, b_{ijk} \in \mathbb{R}$ and $M_j, N, m_i \in \mathbb{N}$, are dense in $C(I_n)$.

In other words given $g \in C(I_n)$ and given $\epsilon > 0$ there is a sum $\psi(x)$ defined as above such that

$$|\psi(x) - g(x)| < \epsilon \quad \forall x \in I_n.$$

Since tanh is a continuous function, it follows that the family of functions defined by equation (4.15) is a linear subspace of $C(I_n)$. To prove that it is dense in $C(I_n)$ we will use the same argument we used for the continuous sigmoidal functions.

This is, called U the set of functions of the form (4.15), we assume that U is not dense in $C(I_n)$. Thus, by the Hahn-Banach theorem there exists a not null bounded linear functional on $C(I_n)$ with the property that it is zero on the closure of U. By the Riesz Representation Theorem, the bounded linear functional can be represented by a Radon measures. Then using the definition of U we will see that this measure must be the zero measure, hence the functional associated with it is null contradicting the hypothesis.

We define

$$h_\lambda(x) = \frac{1}{2}\left[\prod_{k=1}^{m} \tanh(\lambda(w_k(x-b_k)) + \phi) + 1\right]. \tag{4.16}$$

To proceed with the proof as in the case of the proof for continuous sigmoidal functions, we need only to understand to what converges the function

$$\psi_{\lambda_2,\lambda_1}(x) = \sum_{j=1}^{N} \frac{\alpha_j}{2} h_{j\lambda_2}\left(\sum_{i=1}^{n} h_{i\lambda_1}(x)\right) \tag{4.17}$$

when λ_1 and λ_2 tend to infinity, and $h_{i\lambda}$ indicates the processing function related to input i.

Once we have shown that for some choice of the parameters they converge pointwise to step function we can use the same argument we used in the proof of Theorem 4.4.11.

The first step is therefore to study the limit of equation (4.17). Let us focus of the multiplication of tanh in the first layer, given by equation (4.16).

The pointwise limit of $h_\lambda(x)$ for $\lambda \to \infty$ depends on the sign of the limit of the product of tanh, that in turn depends on the sign of $w_k(x - b_k)$ for $k \in \{1, ..., m\}$.

Remark 5. *We remark that for $x \in [0, 1]$, from the limit of equation (4.16) we can obtain the indicator functions of set of the form $x > b$ or $x < b$ for any $b \in \mathbb{R}$. We just have to choose the parameters in such a way that only one of the* tanh *in the multiplication is relevant. Let us define $Z = \{k \in \{1, ..., m\} : w_k(x - b_k) > 0 \quad \forall x \in [0, 1]\}$. If $|Z| = m - 1$, i.e. there is only one $i \in \{1, ..., m\}$ so that its weight are significant it holds that*

$$\lim_{\lambda\to\infty} h_\lambda(x) = \upsilon(x) = \begin{cases} 1 & \textit{if } w_i(x-b_i) > 0 \\ \sigma(2\phi) & \textit{if } w_i(x-b_i) = 0 \\ 0 & \textit{if } w_i(x-b_i) < 0 \end{cases}$$

taking into account that $\sigma(2\phi) = \frac{1}{2}(\tanh(\phi) + 1)$.

Proof of Theorem 4.4.12. Considering Remark 5, the proof of Theorem 4.4.12 is analogous to that of Theorem 4.4.11. □

4.5 Experiments

4.5.1 Datasets

We selected a collection of datasets from the UCI Machine Learning Repository. We only consider classification models in our experiments. However, it is straightforward to apply Newron architectures to regression problems. The description of the datasets is available at the UCI Machine Learning Repository website or the Kaggle website.

We also used 4 synthetic datasets of our creation, composed of 1000 samples with 2 variables generated as random uniforms between −1 and 1 and an equation dividing the space into 2 classes. The 4 equations used are bisector, xor, parabola, and circle.

19 out of 23 datasets are publicly available, either on the UCI Machine Learning Repository website or on the Kaggle website. Here we present a full list of the datasets used, correlated with their shortened and full-length name, and the corresponding webpage where the description and data can be found.

Short name	Full-length name	Webpage
adult	Adult	<UCI_MLR_URL>/adult
australian	Statlog (Australian Credit Approval)	<UCI_MLR_URL>/statlog+(australian+credit+approval)
b-c-w	Breast Cancer Wisconsin	<UCI_MLR_URL>/Breast+Cancer+Wisconsin+(Diagnostic)
car	Car Evaluation	<UCI_MLR_URL>/car+evaluation
cleveland	Heart Disease	<UCI_MLR_URL>/heart+disease
crx	Credit Approval	<UCI_MLR_URL>/credit+approval
diabetes	Diabetes	https://www.kaggle.com/uciml/pima-indians-diabetes-database
german	Statlog (German Credit Data)	<UCI_MLR_URL>/statlog+(german+credit+data)
glass	Glass Identification	<UCI_MLR_URL>/glass+identification
haberman	Haberman's Survival	<UCI_MLR_URL>/haberman%27s+survival
heart	Statlog (Heart)	<UCI_MLR_URL>/statlog+(heart)
hepatitis	Hepatitis	<UCI_MLR_URL>/hepatitis
image	Statlog (Image Segmentation)	<UCI_MLR_URL>/Statlog+(Image+Segmentation)
ionosphere	Ionosphere	<UCI_MLR_URL>/ionosphere
iris	Iris	<UCI_MLR_URL>/iris
monks-1	MONK's Problems	<UCI_MLR_URL>/MONK%27s+Problems
monks-2	MONK's Problems	<UCI_MLR_URL>/MONK%27s+Problems
monks-3	MONK's Problems	<UCI_MLR_URL>/MONK%27s+Problems
sonar	Connectionist Bench	<UCI_MLR_URL>/Connectionist+Bench+(Sonar,+Mines+vs.+Rocks)

Table 4.2. Publicly available datasets, with the short name used in in our work, their full-length name and the webpage where data and description can be found. The UCI_MLR_URL is the following: https://archive.ics.uci.edu/ml/datasets/

The 4 synthetic datasets of our own creation are composed of 1000 samples with 2 variables generated as random uniforms between -1 and 1 and an equation dividing the space into 2 classes. The 4 equations used are:

- bisector: $x_1 > x_2$
- xor: $x_1 > 0 \wedge x_2 > 0$
- parabola: $x_2 < 2x_1^2 - \frac{1}{2}$
- circle $x_1^2 + x_2^2 < \frac{1}{2}$

These datasets are also represented in Figure 4.6.

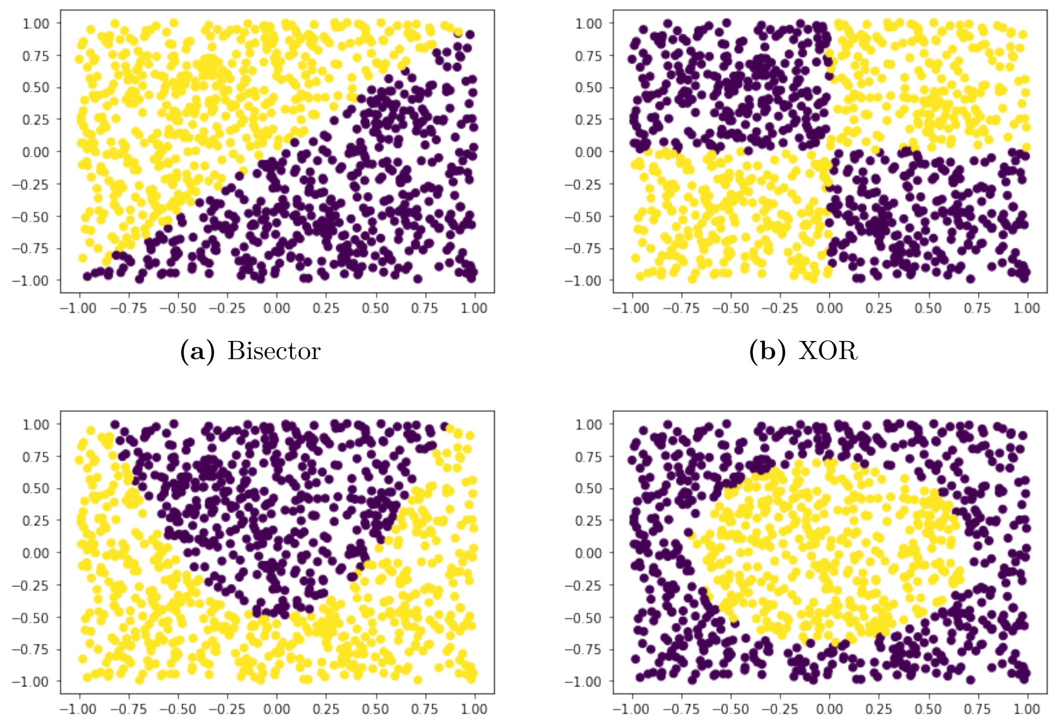
4.5.2 Methods

We run a hyperparameter search to optimize the IAN neural network structure, i.e., depth and number of neurons per layer, for each dataset. We started our search by trying a single neuron, followed by a shallow network and then continued through various DNN configurations. We tested IAN with all three different processing functions. In the tanh-prod case, we set $M = 2$.

Concerning the training of traditional neural networks, we tested the same structures used for Newron, i.e., the same number of layers and neurons. Finally, we also ran a hyperparameter search to find the best combinations in the case of Logistic Regression (LR), Decision Trees (DT), and Gradient Boosting Decision Trees (GBDT).

4.5.3 Results

Table 4.1 presents on each row the datasets used while on the columns the various models. Each cell contains the 95% confidence interval for the accuracy of the model that obtains the best performance.



(a) Bisector **(b)** XOR

(c) Parabola **(d)** Circle

Figure 4.6. The synthetically generated datasets we used to assess the soundness of our methodology.

Results obtained with the new IAN neurons are better than those obtained by DTs and LRs (interpretable) models. Moreover, IAN's results are on par, sometimes better than, results of traditional NNs and GBDT classifiers. These last two methods, though, are not transparent.

Amongst the Heaviside, sigmoid, and tanh-prod cases, we can see that the first one obtains the worst results. The reason may be that it is more challenging to train, despite being the most interpretable among the three cases. tanh-prod instead performs slightly better than sigmoid, being more flexible. Sigmoid, being more straightforward to interpret than tanh-prod, could be a good choice at the expense of a slight decrease in accuracy that remains, however, similar to that of a traditional neural network.

4.5.4 Experimental settings

All code was written in Python Programing Language. In particular, the following libraries were used for the algorithms: tensorflow for neural networks, scikit-learn for Logistic Regression, Decision Trees and Gradient Boosting Decision Trees.

A small exploration was made to determine the best structure of the neural network for each dataset. We used a breadth-first search algorithm defined as follows. We started with a network with just one neuron, we trained it and evaluated its performance. At each step, we can double the number of neurons in each layer except the output one or increase the depth of the network by adding a layer with one neuron. For each new configuration, we build a new structure based on it, initialize it and train it. If the difference between the accuracy achieved by the new structure and that of the previous step is lower than 1%, then a patience parameter is reduced by 1. The patience parameter is initialized as 5 and is passed down from a parent node to its spawned children, so that each node has its own instance of it. When patience reach 0, that configuration will not spawn new ones.

Before the neural network initialization, a random seed was set in order to reproduce the same results. As for the initialization of IAN, the weights w are initialised using the glorot uniform. For the biases b of the first layer a uniform between the minimum and the maximum of each feature was used, while for the following layers a uniform between the minimum and the maximum possible output from the neurons of the previous layer was used.

For the network training, Adam with a learning rate equal to 0.1 was used as optimization algorithm. The loss used is the binary or categorical crossentropy, depending on the number of classes in the dataset. In the calculation of the loss, the weight of each class is also taken into account, which is inversely proportional to the number of samples of that class in the training set. The maximum number of epochs for training has been fixed at 10000. To stop the training, an early stopping method was used based on the loss calculated on the train. The patience of early stopping is 250 epochs, with the variation that in these epochs the loss must decrease by at least 0.01. Not using a validation dataset may have led to overfitting of some structures, so in future work we may evaluate the performance when using early stopping based on a validation loss. The batch size was fixed at 128 and the training was performed on CPU or GPU depending on which was faster considering the amount of data. The Heaviside was trained as if its derivative was the same as the sigmoid.

For Decision Trees (DT) and Gradient Boosting Decision Trees (GBDT), an optimisation of the hyperparameters was carried out, in particular for min_samples_split (between 2 and 40) and min_samples_leaf (between 1 and 20). For GBDT, 1000 estimators were used, while for DT the class_weight parameter was set. For the rest of the parameters, we kept the default values of the python sklearn library.

4.5.5 Examples

Circle dataset example

In order to first validate our ideas, we show what we obtained by applying a single neuron using multiplication of 2 tanh in the case of our custom dataset circle.

In Figure 4.7 we can see how the multiplication of tanh has converged to two bells centred in 0, while α_1 and α_2 have gone to 30. According to the IANinterpretation method, values below 30 correspond to an activation function output of 0, while it is 1 for values above 38. In the middle range, the prediction is more uncertain. Combining this data with the previous prediction, we can conclude that we need the sum of the two values output by the two processing functions to be greater than 38 to have a prediction of class 1. Therefore, if one of the two inputs is 0 (output 30), it is enough for the other to be between -0.65 and 0.65 (output greater than 8). Otherwise, we may need an output of at least 19 from both outputs, corresponding to input values between -0.5 and 0.5, i.e., the area covered by the circle.

Heart dataset - Heaviside IAN

The Statlog Heart dataset is composed of 270 samples and 13 variables of medical relevance. The dependent variable is whether or not the patient suffers from heart disease. In Figure 4.8 you can find the network based on Heaviside IAN trained on the heart dataset. Only the inputs with a relevant contribution to the output are

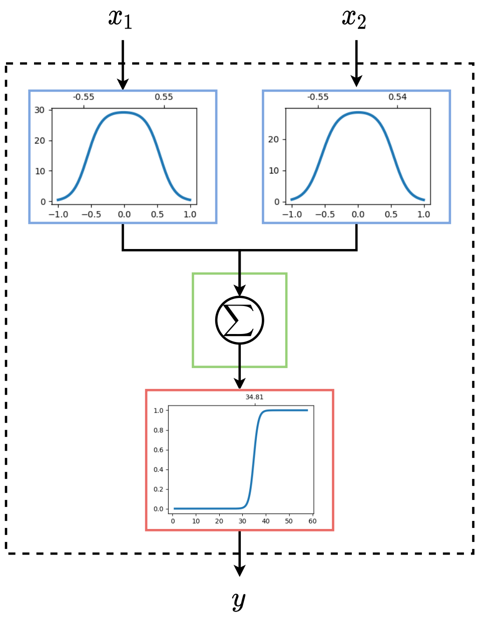


Figure 4.7. tanh-prod Neural Network trained on the circle dataset. The figure follows the color convention used for NEWRON in Figure 4.3. x_1 and x_2 are the inputs of the network and y is the output. The processing and activation functions are plotted with input on the x-axis and output on the y-axis. Coordinates of the inflection points are indicated above the plots.

shown. From now on, we will indicate with $R_{k,j,i}$ the rule related to the processing function corresponding to the i-th input, of the j-th neuron, of the k-th layer. From the first neuron of the first layer we can easily retrieve the following rules: $R_{1,1,1} = x_1 \leq 54.29, R_{1,1,3} = x_3 \leq 3.44, R_{1,1,4} = x_4 \leq 123.99, R_{1,1,5} = x_5 \geq 369,01, R_{1,1,9} = x_9 \leq 0.48, R_{1,1,10} = x_{10} \leq 1.22, R_{1,1,11} = x_{11} \leq 1.44, R_{1,1,12} = x_{12} \leq 0.52, R_{1,1,13} = x_{13} \leq 6.26$. The second neuron of the first layer is not shown for lack of space, but its obtained rules are $R_{1,2,2} = x_2 \geq 0.79, R_{1,2,3} = x_3 \geq 3.59, R_{1,2,4} = x_4 \geq 99.95, R_{1,2,5} = x_5 \geq 253.97, R_{1,2,8} = x_8 \leq 97.48, R_{1,2,9} = x_9 \leq 0.04, R_{1,2,10} = x_{10} \geq 2.56, R_{1,2,11} = x_{11} \geq 1.53, R_{1,2,12} = x_{12} \geq 0.52, R_{1,2,13} = x_{13} \geq 5.47$. Moreover, input x_7 gives always 1, so this must be taken into consideration in the next layer.

Moving on to the second layer, we can see in the first neuron that the second input is irrelevant, since the Heaviside is constant. The first processing function activates if it receives an input that is greater or equal to 2.99. Given that the input can only be an integer, we need at least 3 of the rules obtained for the first neuron of the first layer to be true: $R_{2,1,1} = 3 - of - \{R_{1,1,i}\}$. Following the same line of reasoning, in the second neuron of the second layer we see that we get $R_{2,2,1} = 5 - of - \{\neg R_{1,1,i}\}$ and $R_{2,2,2} = 5 - of - \{R_{1,2,i}\}$ (5 and not 6 because of x_7 processing function).

In the last layer, the first processing function has an activation of around 2.5 if it receives an input that's less than 1.17. This can happen only if $R_{2,1,1}$ does not activate, so we can say: $R_{3,1,1} = \neg R_{2,1,1} = 7 - of - \{\neg R_{1,1,i}\}$. The second processing function gives a value of around -2.5 only if it gets an input less than 0.99, so only if the second neuron of the second layer does not activate. This means that $R_{2,2,1}$ and $R_{2,2,2}$ must be both false at the same time, so we get $R_{3,1,2} = \neg R_{2,2,1} \wedge \neg R_{2,2,2} = 5 - of - \{R_{1,1,i}\} \wedge 6 - of - \{\neg R_{1,2,i}\}$. Now there are 4 cases for the sum, i.e. the combinations of the 2 activations: $\{0 + 0, 2.5 + 0, 0 - 2.5, 2.5 - 2.5\} = \{-2.5, 0, 2.5\}$.

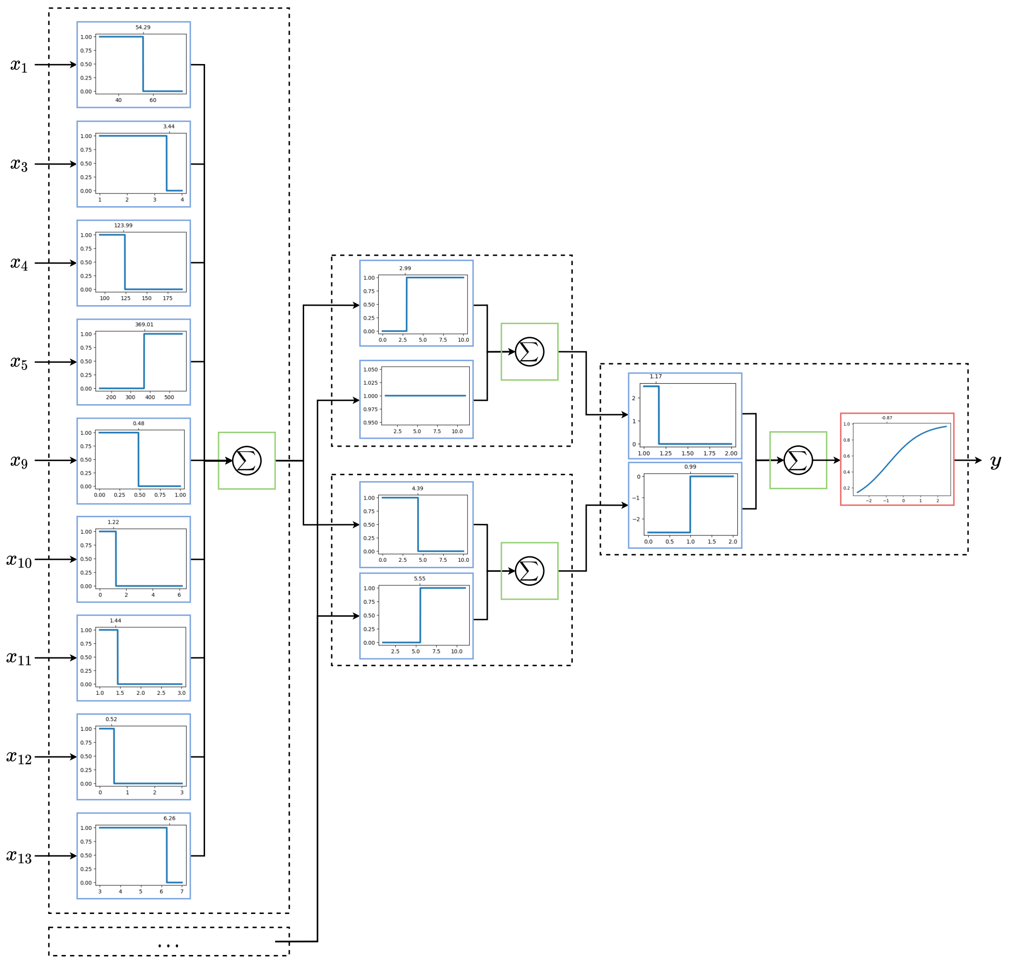


Figure 4.8. The Heaviside IAN Network trained on the heart dataset. The Figure follows the color convention used for Newron.

Given that both have around the same value for the α parameter, the set is reduced to two cases. Looking at the processing function, we can see that is increasing with respect to the input, so since α_1 is positive, we can say that rule $R_{3,1,1}$ is correlated to class 1, while rule $R_{3,1,2}$, having a negative α_2, has an opposite correlation. Looking at its values, we can see that for both 0 and 2.5 inputs, the activation function gives an output greater than 0.5. If we consider this as a threshold, we can say that only for an input of -2.5 we get class 0 as prediction. This happens only if $R_{3,1,2}$ is true and $R_{3,1,1}$ is false. Summarizing we get $R_0 = R_{3,1,2} \wedge \neg R_{3,1,1} = 5 - of - \{R_{1,1,i}\} \wedge 6 - of - \{\neg R_{1,2,i}\} \wedge 3 - of - \{R_{1,1,i}\} = 5 - of - \{R_{1,1,i}\} \wedge 6 - of - \{\neg R_{1,2,i}\}$, so that we can say "if R_0 then predicted class is 0, otherwise is 1".

Although we are not competent to analyse the above results from a medical perspective, it is interesting to note for example that the variables x_1 and x_4, representing age and resting blood pressure respectively, are positively correlated with the presence of a heart problem.

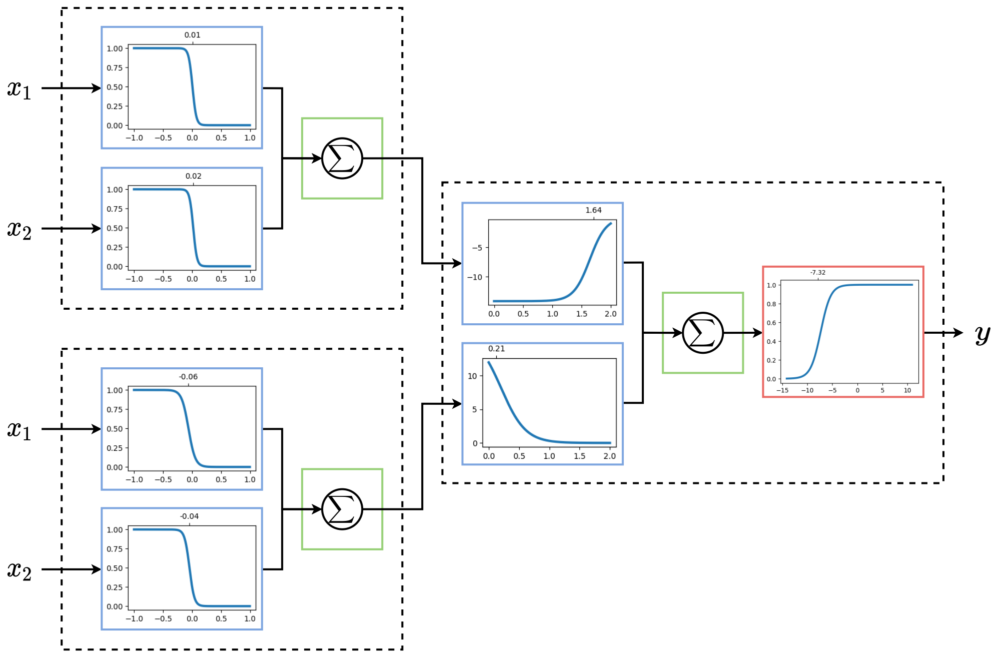


Figure 4.9. The sigmoid IAN Network trained on the xor dataset. The Figure follows the color convention used for NEWRON.

Xor - sigmoid IAN

Our custom xor dataset divides the 2D plane in quadrants, with the opposites having the same label.

The network based on sigmoid IAN trained on xor dataset is represented in Figure 4.9. As we can see, all the processing functions of the first layer converged to nearly the same shape: a steep inverted sigmoid centered in 0. Therefore, we can say the rules obtained are $R_{1,1,1} = R_{1,2,1} = x_1 \leq 0$ and $R_{1,1,2} = R_{1,2,2} = x_2 \leq 0$. In the last layer, the first processing function has a value of about -15 for inputs in $[0, 1]$, then it starts growing slowly to reach almost 0 for an input of 2. This tells us that it doesn't have an activation if both rules of the first neuron are true, so if $x_1 \leq 0 \wedge x_2 \leq 0$. On the other hand, the second processing function has no activation if its input greater than 1, that happens for example if we have a clear activation from at least one of the inputs in the second neuron of the first layer. So looking at it the opposite way, we need both those rules to be false $(x_1 > 0 \wedge x_2 > 0)$ to have an activation of 12.5. The activation function is increasing with respect to the input, and to get a clear class 1 prediction, we need the input to be at least -5. Considering if the processing functions could give only $\{-15, 0\}$ and $\{12.5, 0\}$ values, just in the case we got -15 from the first one and 0 from the second one ot would give us a clear class 0 prediction. This happens only if $\neg(x_1 \leq 0 \wedge x_2 \leq 0) = x_1 > 0 \vee x_2 > 0$ and $\neg(x_1 > 0 \wedge x_2 > 0) = x_1 \leq 0 \vee x_2 \leq 0$, that can be summarised $(x_1 > 0 \vee x_2 > 0) \wedge (x_1 \leq 0 \vee x_2 \leq 0) = (x_1 > 0 \wedge x_2 \leq 0) \wedge (x_1 \leq 0 \vee x_2 > 0)$. Since this rule describes the opposite to xor, for class 1 we get the exclusive or logical operation.

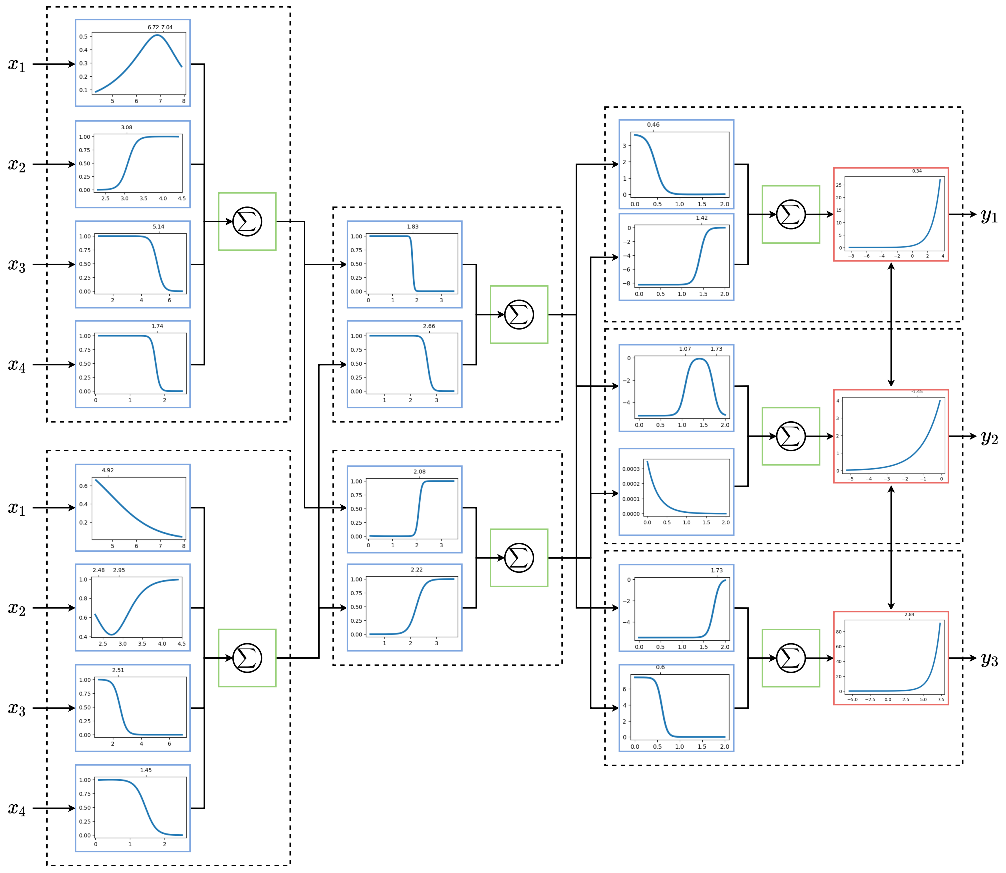


Figure 4.10. The tanh-prod IAN Network trained on the iris dataset. The Figure follows the color convention used for NEWRON.

Iris dataset - hyperbolic tangents IAN

A dataset widely used as a benchmark in the field of machine learning is the Iris dataset. This contains 150 samples, divided into 3 classes (setosa, versicolor and virginica) each representing a type of plant, while the 4 attributes represent in order sepal length and width and petal length and width.

In Figure 4.10 you can see the final composition of the network generated with the tanh-prod2 IAN neuron.

Considering the first neuron of the first layer, we see that it generates the following fuzzy rules: $R_{1,1,2} = x_2 > 3.08$ (sepal width), $R_{1,1,3} = x_3 < 5.14$ (petal length) and $R_{1,1,4} = x_4 < 1.74$ (petal width). For the first attribute (sepal length) it does not generate a clear rule, but forms a bell shape, reaching a maximum of 0.5. This tells us that x_1 is less relevant than the other attributes, since, unlike the other processing functions, it does not reach 1. The second neuron has an inverse linear activation for the first attribute, starting at 0.7 and reaching almost 0. The second attribute also has a peculiar activation, with an inverse bell around 2.8 and a minimum value of 0.4. The third and fourth attributes have clearer activations, such as $R_{1,2,3} = x_3 < 2.51$ and $R_{1,2,4} = x_4 < 1.45$.

The fact that petal length and width are the ones with the clearest activations and with those specific thresholds are in line with what has previously been identified on the Iris dataset by other algorithms.

We denote by $y_{k,j}$ the output of the j-th neuron of the k-th layer. Moving on to the second layer, the first neuron generates the rules "if $y_{1,1} < 1.83$" and "if $y_{1,2} < 2.66$", while the second one generates "if $y_{2,1} > 2.08$" and "if $y_{2,2} > 2.22$". Combined with what we know about the previous layer, we can deduce the following: $y_{1,1}$ is less than 1.83 only if the sum of the input activation functions is less than 1.83, which only happens if no more than one of the last three rules is activated $(0 + 1 + 0 < 1.83)$, while the first one, even taking its maximum value, is discriminative only when the input of one of the other rules is close to the decision threshold $(0.5+1+0+0 < 1.83$, while $0.5 + 1 + 0.5 + 0 > 1.83)$. For $y_{1,2} < 2.66$, there are more cases. We can divide the second processing function of the second neuron of the first layer in two intervals: one for which $x_2 < 3.2$ and the other when $x_2 \geq 3.2$. In the first interval, the processing function gives a value that is less than 0.66, greater in the second one. With this, we can say that $y_{1,2} < 2.66$ even if $R_{1,2,3}$ and $R_{1,2,4}$ activates, if $x_2 < 3.2$ and x_1 is near its maximum.

In the second neuron of the second layer, the first processing function is nearly the exact opposite to that of the other neuron; we need at least two of $R_{1,1,2}$, $R_{1,1,3}$ or $R_{1,1,4}$ to be true, while $R_{1,1,1}$ still doesn't have much effect. The second processing function gives us $y_{1,2} > 2.22$. Considering that the minimum for the processing function related to x_2 is 0.4, we may need both rules $R_{1,2,3}$ and $R_{1,2,4}$ to be true to exceed the threshold, or just one of them active and x_1 to take on a low value and x_2 to be a high value.

For the last layer, remember that in this case since there are more than 2 classes, a softmax function is used to calculate the output probability, hence the arrows in the figure that join the layers of the last layer.

For the first output neuron, in order to obtain a clear activation, we need the first input to be less than 0.46 and the second greater than 1.42. This is because the α_i are 3 and -8, and the output activation function starts to have an activation for values greater than -2. This means that the first neuron of the second layer should hardly activate at all, while the other should activate almost completely. Considering the thresholds for $y_{1,1}$ and $y_{1,2}$, we need the first to be greater than 2.08 and the other to be greater than 2.66. So $R_{3,1,1} = 2 - of - \{x_2 > 3.08, x_3 < 5.14, x_4 < 1.74\}$. For $R_{3,1,2}$ is more tricky to get a clear decision rule, but we can say that we may need both $R_{1,2,3}$ and $R_{1,2,4}$ to be true and $x_2 \geq 3.2$. If $x_2 < 3.2$, we need x_1 to not be near its maximum value. If just one of those two rules is true, we need $x_2 < 3.2$ and x_1 near 4, or $x_2 > 3.2$ but with a (nearly) direct correlation with x_1, such that the more x_1 increases, the same does x_2.

In the second output neuron, the second processing function is negligible, while the first one forms a bell shape between 1 and 2. This means that it basically captures when $y_{2,1}$ has a value of approximately 1.5, so when the decision is not clear. This is what gives this neuron maximum activation.

In the third and last output layer, since the first processing function has a negative α parameter and the activation function is increasing with respect to the input, we want it to output 0, and this requires maximum activation for the first neuron of the second layer. Regarding the second processing function, we want it to output 8, so we need nearly no activation from the second neuron of the second layer. So we need the first neuron of the first layer to output a value lower than 1.83 and the second neuron to output a value lower than 2.22. This means that no more than one rule $R_{1,1,i}$ needs to be active and at most two rules of $R_{1,2,i}$ need to be true.

We can conclude by saying that both neurons of the first layer are positively correlated with class 1, while they are negatively correlated with class 3. This means that low values of x_3 and x_4, or high values of x_2 increase the probability of a sample to belong to class 1, while x_1 has almost no effect. For class 2, what we can say is that it correlates with a non-maximum activation of both neurons of the first layer, meaning that it captures those cases in which the prediction of one of the other classes is uncertain.

4.5.6 Current limitations

The extraction of proper rules from the network can be harrowing; in the Heaviside case, they might be too long in the sigmoid and tanh-prod cases because their simplicity depends on the final value parameters. Nevertheless, methods of regularization during training or additional Rule Extraction methods may help to simplify interpretability. We defer the study of regularization to future works.

Also, we have not compared Newron against state-of-the-art Deep Learning models for tabular data, as our main goal was to show that our formulation was more suitable than traditional neurons compared to "traditional" interpretable models. Comparisons with more advanced solutions for tabular data will be the subject of future work.

4.6 Conclusions and Future Work

We have introduced the concept of a generalized neuron and proposed three different specializations, along with the corresponding method to interpret the behavior of the network. Also, in cases where from the network we cannot devise exact rules (e.g., in the sigmoid and tanh-prod cases), the structure of the neuron and the parameters allow the visualization of its behavior. Indeed, for every input, we apply the nonlinearity operation before the aggregation reducing it to a one-dimensional space allowing the analysis of each input separately. Through universal approximation theorems, we have proved that the new structure retains the same expressive power as a standard neural network. In future studies we will investigate more in detail the expressiveness of IAN based models with respect to the number of layers or neurons in arbitrarily deep but width-limited networks and arbitrarily wide but depth-limited networks. Experiments conducted on both real and synthetic datasets illustrate how our framework can outperform traditional interpretable models, Decision Trees, and Logistic Regression, and achieve similar or superior performance to standard neural networks. In the future, we will investigate the influence of hyper-parameters (network depth, number of neurons, processing functions) and initialization on the model quality. Also, we will refine the analysis of the tanh-prod case as the number of tanh increases. In addition, we will investigate IAN with additional processing functions, such as ReLU and SeLU. Finally, we will extend this method to other neural models, such as Recurrent, Convolutional and Graph Neural Networks. Although we have not yet defined exactly how to extend to the other cases, the general idea remains the same: avoid linear combinations, instead apply a function to each input and then aggregate the results. Since CNNs are in fact a special case of Fully-connected NNs with certain weights fixed and/or shared, our neuron would already be applicable to images, but the interpretation for this case will require more investigation.

Chapter 5

A topological description of loss surfaces via Betti Numbers characterization

In recent years, the pursuit for the theoretical foundations of Machine Learning (ML) and Deep Learning (DL) models has gained more and more attention as the research community decided to delve into the reasons why these models can achieve outstanding performance in different application fields (Koren et al., 2009; Scarselli et al., 2008; Krizhevsky et al., 2017; Devlin et al., 2018; Radford et al., 2018). On the one hand, as the automated decisions provided by these algorithms can have a relevant impact on people's lives, their behaviour has to be aligned with the values and principles of individuals and society. This demands designing automated methods we can trust, fulfilling the requirements of fairness, robustness, privacy, and explainability (Oneto et al., 2022). On the other hand, a wide range of tools arose from different areas have been taken into account to give proper explanations to the behaviour of ML and DL models according to different mathematical aspects, such as the gradient descent dynamics (Maennel et al., 2018) (Williams et al., 2019) (Goodfellow et al., 2014), the role of the activation functions (Ramachandran et al., 2017), the importance of the number of layers (Bianchini and Scarselli, 2014). From the theoretical point of view, the characterization of the loss function to be minimized is crucial, as the whole training efficiency relies on its shape, which in turn depends on the network architecture. Several works have already dealt with the analysis of the surface of the loss function, identifying conditions for the presence (or absence) of spurious valleys in a theoretical (Venturi et al., 2018) or empirical-driven way (Safran and Shamir, 2018), pointing out the role of saddle points in slowing down the learning (Dauphin et al., 2014), and giving hints on the topological structure of the loss for networks with different activation functions (Freeman and Bruna, 2016; Nguyen and Hein, 2017).

Our contribution fits into the latter line of research, as we give a characterization of the complexity of loss functions by a topology argumentation. More precisely, given a layered neural network $\mathcal{N}$ and a loss function $\mathcal{L}_{\mathcal{N}}$ computed on some train data, we will measure the complexity of $\mathcal{L}_{\mathcal{N}}$ by the topological complexity of the set $S_{\mathcal{N}} = \{\theta | \mathcal{L}_{\mathcal{N}}(\theta) \leq c\}$. Such an approach is natural, since $S_{\mathcal{N}}$, observed at each level c, provides the form of the loss function: for example, if $\mathcal{L}_{\mathcal{N}}$ has k isolated minima, then $S_{\mathcal{N}}$ has k disconnected regions for some small c.

In the chapter, we will provide a bound on the sum of Betti numbers (Bredon, 2013) of the set $S_{\mathcal{N}}$. In algebraic topology, Betti numbers distinguish spaces with different topological properties. More precisely, for any subset $S \subset \mathbb{R}^n$, there exist n Betti numbers $b_i(S), 0 \leq i \leq n-1$, Intuitively, the first Betti number $b_0(S)$ is the number of connected components of the set S, which in the case $S = S_{\mathcal{N}}$ correspond to the number of basins of attraction of the loss function. The i-th Betti number $b_i(S)$ counts the number of $(i+1)$-dimensional holes in S, which measures the tortuosity of the error function on a given level.

The upper bound is derived for feedforward neural networks with different numbers of layers, number of neurons per layer, and number of training samples. Moreover, we consider networks with skip connections, such as ResNet, and MSE loss functions with or without a regularization. Finally, we treat networks with Pfaffian activation functions. The class of Pfaffian maps is wide and includes most of the functions, such a large subset of the functions with continuous derivatives commonly used in engineering and computer science applications, such as hyperbolic tangent, logistic sigmoid and polynomials and their composition [1]. Actually, most of the elementary functions are Pfaffian, e.g., the exponential and the trigonometric functions, which is a wide class. The concept of Pfaffian functions was initially introduced by Khovanskii (Khovanskii, 1991) and Gabrielov and Vorobjov formulated this concept of Pfaffian complexity or format in their work Gabrielov and Vorobjov (1995). We will define these concepts formally later in the Preliminaries section.

It is worth mentioning that this work takes inspiration from Bianchini and Scarselli (2014), where a similar topological argumentation has been used to study the complexity of the functions implemented by a neural network, showing that deep networks can implement more complex functions than shallow ones using the same number of parameters. Thus, intuitively, while the results in Bianchini and Scarselli (2014) are about the network function, namely what a network can do, the results in this paper are about the error function, namely how hard the optimization problem is.

Our main contributions are summarized as follows:

- we derive the Pfaffian format of common loss function, i.e. the Mean Square Error (MSE) loss function and the Binary Cross-Entropy (BCE) loss function with respect to the training of feedforward perceptron networks; the Pfaffian format is computed with respect to the weights, and involves the knowledge of the Pfaffian format of Pfaffian activation function;

- consequently, bounds on the complexity of said loss functions are found in terms of Betti numbers characterization; multiple bounds are described independently in terms of the different parameters of the problem (i.e. number of layers, number of neurons per layer, size of the training dataset, total number of learnable parameters in the network). Specifically, we show (coherently with the literature and common knowledge) that the complexity of the loss function increases super-exponentially with the number of layers or neurons per layer, and exponentially with the number of training samples. Moreover, considerations around the possible implementation of regularizers or different architectures are drawn to give a wider perspective.

The chapter is organized as follows. In Section 5.1, we briefly review the literature on the characterization of the loss surface and topological tools used to evaluate the

[1]For sake of simplicity, we do not consider networks with ReLU activation functions, which are not Pfaffian. The results of this work can also be extended to ReLU and, more generally, piece-wise polynomials, using a measure of complexity based on the number of disconnected components of the set $S_{\mathcal{N}}$. For example, such an argument has been used to estimate the Vapnick Chernovenkis dimension of feedforward neural networks (Bartlett et al., 1998b). However, the estimation requires a different technique and a duplication of our theorems, which we prefer to skip here for simplicity.

computational power and generalization ability of feedforward networks. In Section 5.2, some notations and preliminary definitions are introduced, while in Section 5.3, the main results proposed in this work are presented. To ensure easy text reading, which allows for capturing the main contents, all proofs are collected in Section 5.4. Finally, a discussion of the theoretical outcomes of our study and some conclusions are reported in Sections 5.5 respectively.

5.1 Related work

The characterization of the loss landscape has gained more and more attention in the last few years; attempts to provide a satisfying description have been made through several approaches. In Choromanska et al. (2015a,b), the spin-glass theory is exploited to quantify, in probability, the presence of local minima and saddle points. Unfortunately, the connection between the loss function of neural networks and the Hamiltonian of the spherical spin-glass models relies on several possibly unrealistic assumptions. Yet, the empirical evidence suggests that it may also exist under mild conditions. In Dauphin et al. (2014), a method that can rapidly escape high dimensional saddle points, unlike gradient descent and quasi-Newton methods, is proposed. Based on results from statistical physics, random matrix theory, neural network theory, and empirical evidence, it is argued that the major difficulty in using local optimization methods originates from the proliferation of saddle points instead of local minima, especially in high dimensional problems of practical interest. Saddle points are surrounded by plateaus that can dramatically slow down the learning process. In Venturi et al. (2019), the topological property of the loss function defined as *presence or absence of spurious valleys* (defined as connected components of the sub-level sets that do not contain a global minima) is addressed for one-hidden layer neural networks, providing the following contributions: 1) the Empirical Risk Minimization loss for any continuous activation function and the expected value loss with polynomial activations do not exhibit spurious valleys as long as the network is sufficiently over-parametrised. 2)For non-polynomial non-negative activations, among which ReLU networks are included, for any hidden width, there exists a data distribution which yields spurious valleys with positive measure, whose value is arbitrarily far from the one of the global.

Based on a computer-driven empirical proof, similar results are also shown in Safran and Shamir (2018).

In Freeman and Bruna (2019), the loss surface is studied in terms of level sets, and it is shown that the landscape of the loss functions of deep networks with linear activation functions is significantly different from that exploiting half-rectified ones:

in the former case, the level set can have disconnected components. The level sets are connected without nonlinearities, whereas the level set can be disconnected. Finally, a comprehensive research survey focused on analysing the loss landscape can be found in Berner et al. (2021).

In algebraic topology, Betti numbers are used to distinguish spaces with different topological properties. Betti numbers have been exploited to give a topological description of the complexity of he function implemented by neural networks with Pfaffian activation functions and to describe their generalization capabilities. More specifically, in Karpinski and Macintyre (1997), by such a technique, bounds on the VC dimension of feedforward neural networks are provided; this work has been later extended to Recurrent and Graph Neural Networks (Scarselli et al., 2018). In Bianchini and Scarselli (2014), bounds on the sum of Betti numbers are provided to describe the complexity of the map implemented by feedforward networks with

Pfaffian activation functions. In Naitzat et al. (2022), the relative efficacy of ReLUs over traditional sigmoidal activations is justified based on the different speeds with which they change the topology of a data set representing two classes of objects in a binary classification problem. Specifically, they examine how the topology of the dataset's manifold changes as it moves through the layers of a well-trained neural network. This dataset is viewed as a combination of two components: the first component represents the manifold of elements from the first class, and the second component encompasses the elements from the second class. This investigation is done considering a well-trained neural network, i.e., one with perfect accuracy on its training set and a near-zero generalization error. A ReLU-activated neural network (neither a homeomorphism nor Pfaffian) can sharply reduce Betti numbers of the two components, but not a sigmoidal-activated one (which is a homomorphism). Reducing the Betti numbers means that the neural network simplify the structure of the dataset by reducing the number of connected components, holes, or voids within the data's manifold. This reduction suggests that the network is effectively simplifying or transforming the dataset's topology, making it more amenable for analysis and classification. Finally, their research suggests that when dealing with higher topological complexity data (meaning the data has more intricate or complex shapes and relationships), we need neural networks with greater depth (more layers) to adequately capture and understand these complexities.

5.2 Preliminaries

In this section, we introduce the notation, basic concepts, and definitions, which will be functional to the subsequent description of the main results. In the following, we'll deal exclusively with feedforward perceptron neural networks, whose definition is introduced as follows.

Feedforward neural networks — Let $\theta = \{\tilde{W}^1, b^1, \dots, \tilde{W}^L, b^L\}$ be the set of the trainable network parameters, where $\tilde{W}^l \in \mathbb{R}^{n_l \times n_{l-1}}$, $b^l \in \mathbb{R}^{n_l \times 1}$ and $l = 1, \dots, L$ identifies each layer, $n_0, \dots, n_L \in \mathbb{N}$ denote the number of neurons per layer. In the following without loss of generality, we assume that the network has a single output, i.e., $n_L = 1$. Notice that the last assumption is not necessary to demonstrate the Pfaffian nature of the loss function. However, its inclusion significantly simplifies the subsequent calculations involved in the analysis. The total number of parameters is $\tilde{n} = \sum_{l=1}^{L} n_l n_{l-1}$.

Let $x \in \mathbb{R}^{n_0}$ be the input to the neural network. The function implemented by the layered network is $f_\theta(x) : \mathbb{R}^{n_0} \to \mathbb{R}$, where $f_\theta(x) = g(\tilde{W}^L \sigma(\tilde{W}^{L-1} \cdots \sigma(\tilde{W}^1 x + b^1)) \cdots + b^{L-1}) + b^L)$, σ is the hidden layer activation function and g is the activation function of the output neuron.

To simplify the notation, we will aggregate the weights and bias from the same layer into a single matrix, the *augmented weight matrix* $W^l = [b^l, \tilde{W}^l]$; moreover, we will denote by z^l the output of the l-th layer and by a^l the neuron activations at the same layer. Thus, we have

$$z^0 = \begin{bmatrix} 1 \\ x \end{bmatrix}, \qquad a^l = W^l z^{l-1}, \qquad z^l = \begin{bmatrix} 1 \\ \sigma(a^l) \end{bmatrix}, \qquad \text{for } 1 \le l \le L\,,$$

and $f_\theta(x) = g(a^L)$.

Loss functions — Let $D = (x_i, y_i)_{i=1}^m$ be a set of training data, with $x_i \in \mathbb{R}^{n_0}$ and $y_i \in \mathbb{R}$. Let $\mathcal{L}$ be a generic loss to be minimised over the parameters θ. The empirical risk of loss (or cost) function is defined as the average of the per-sample contributions, where each sample contribution measures the error between the network output and the target value for that sample.

$$\mathcal{L}(\theta, D) = \frac{1}{m}\sum_{i=1}^{m} loss(f_\theta(x_i), y_i),$$

being y_i the target for the i-the pattern x_i. In this work, we will study the topological complexity of the landscape of the Mean Square Error (MSE) and Binary Cross-Entropy (BCE) loss functions, which are defined as

$$\begin{aligned}\mathcal{L}_{\text{MSE}}(\theta, D) &= \frac{1}{m}\sum_{i=1}^{m}(y_i - f_\theta(x_i))^2,\\ \mathcal{L}_{\text{BCE}}(\theta, D) &= \sum_{i=1}^{m} -y_i \log(f_\theta(x_i)) - (1-y_i)\log(1 - f_\theta(x_i)).\end{aligned}$$

In the following we will remove the dependency on the dataset D in the notation of empirical risk. Our focus will center solely on its reliance on the network parameters θ. Indeed we want to study the topological complexity of the sublevel set of the empirical risk as a function of the parameters of the netwrok, considering the dataset samples as fixed. Unless specified otherwise, the notation $\mathcal{L}(\theta)$ in the following will represent the empirical risk corresponding to the loss function $\mathcal{L}$ applied to a dataset containing m pairs (x_i, y_i)

In general, we consider the targets to be real numbers. However, for classification problems where the Binary Cross-Entropy (BCE) loss is used, each target y_i is either 0 or 1.

Moreover, we consider a regularized form for the objective function that can be expressed as

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \lambda\Omega(\theta),$$

where $\Omega(\theta)$ is a regularization term, for instance, if $\Omega(\theta) = \frac{1}{2}\|w\|_2^2$.

Pfaffian Functions — Pfaffian functions (Khovanskii, 1991) are analytic functions satisfying triangular systems of first-order partial differential equations with polynomial coefficients.

For this kind of functions an analogous of the Bézout theorem holds. The classical Bézout theorem states that the number of complex solutions of a set of k polynomial equations in k unknowns can be estimated in terms of their degrees (it equals the product of the degrees).

For a wide class of real transcendental equations (including all real algebraic ones) the number of solutions of a set of k such equations in k real unknowns if finite and can be explicitly estimated in terms of the " complexity" of the equations which leads to the version of Bézout theorem for Pfaffian curves and Pfaffian manifold (Khovanskii, 1991).

The class of Pfaffian functions includes a wide variety of known functions, e.g. the elementary functions, including exponential, logarithm, tangent, and their combinations

(Gabrielov and Vorobjov, 2004). Interestingly, many common activation functions used in neural networks, such as the sigmoid function and hyperbolic tangent, can be classified as Pfaffian functions. Intuitively, a function is Pfaffian if its derivatives can be defined in terms of polynomials of the original function and/or a chain of other Pfaffian functions. Formally, we can state the following definition.

Definition 5.2.1. A *Pfaffian chain* of order $\ell \geq 0$ and degree $\alpha \geq 1$, in an open domain $U \subseteq \mathbb{R}^n$, is a sequence of real analytic functions $f_1, f_2, \dots, f_\ell$, defined on U, satisfying the differential equations

$$\frac{\partial f_i}{\partial x_j}(x) = P_{ij}(x, f_1(x), \dots, f_i(x)),$$

for $1 \leq i, j \leq \ell$ and $x = (x_1, \dots, x_n) \in U$. Here the $P_{ij}(x, y_1, \dots, y_i)$ are polynomials in the $n + i$ variables $x_1, \dots, x_n, y_1, \dots, y_i$ of degrees not exceeding α.

Definition 5.2.2. Let $(f_1, \dots, f_\ell)$ be a Pfaffian chain of length ℓ and degree α, and let U be its domain. A function f defined on U is called a Pfaffian function of degree β in the chain $(f_1, \dots, f_\ell)$ if there exists a polynomial P in $n + \ell$ variables, of degree at most β, such that $f(x) = P(x, f_1(x), \dots, f_\ell(x))$, $\forall x \in U$. The triple (α, β, ℓ) is called the format of f.

First, the polynomial P_{ij} itself may explicitly depend on x. Second, even if P_{ij} doesn't directly depend on x, it can still depend on x indirectly through the function $f_1(x)$. We say that the polynomial P_{ij} depends directly on x if occurrences of x are not of the type $f_i(x)$. For example, consider the function $f(x) = \arctan(x)$. It is a Pfaffian function belonging to case 2, as it can be represented by the chain (f_1, f_2), where $f_2(x) = \arctan(x)$ and $f_1(x) = (1+x^2)^{-1}$. In this case, $\frac{\partial f_2}{\partial x} = P_1(x, f_1) = f_1(x)$, and $\frac{\partial f_1}{\partial x} = P_2(x, f_1) = x f_1(x)^2$. This explicit dependence on x in the polynomial P_2 classifies it under case 2. On the other hand, the tanh function is a Pfaffian function falling under case 1. It can be represented by the chain containing only $f_1(x) = \tanh(x)$, and $\frac{\partial f_1(x)}{\partial x} = P_1(x) = 1 - f_1(x)^2$. In this case, there is no explicit dependence on x in the polynomial P_1. This distinction will be crucial to assess the right computations in the statement of Theorem 5.3.1.

Let us now introduce the notion of Pfaffian variety and semi-Pfaffian variety.

Definition 5.2.3. The set $V \subset U$ is a Pfaffian variety if there are Pfaffian functions $p_1, \dots, p_r$ in the chain $(\mathrm{f}_1, \dots, \mathrm{f}_\ell)$ such that $V = \{x \in U : p_1(x) = \dots = p_r(x) = 0\}$.

Definition 5.2.4. A basic semi-Pfaffian set S on the variety V is a subset of V defined by a set of sign conditions (inequalities or equalities) based on the Pfaffian functions $p_1, \dots p_s$ in the chain $(f_, \dots, f_\ell)$ such that $S = \{x \in V : p_1(x)\varepsilon_1 0 \ \ \& \ \ \cdots \ \ \& \ p_s(x)\varepsilon_s 0\}$, where $\varepsilon_1, \dots, \varepsilon_s$ are any comparison operator among $\{<; >; \leq, \geq; =\}$.

As outlined in the introduction, our focus in this work is directed towards exploring the complexity of the topological space defined by the sublevel set of the empirical risk related to the loss function $\mathcal{L}$: $S = \{\theta : \mathcal{L}(\theta) \leq c\}$. This set represents the collection of parameter values θ for which the empirical risk of the loss function is less than or equal to a constant c. When the empirical risk of the loss function is a Pfaffian function with respect to the parameter of the network, this set is exactly a Pfaffian semi-variety.

Level curves or basins of attractions can often be described in terms of Pfaffian varieties, whose complexity can be measured through the characterization of its Betti numbers (Bianchini and Scarselli, 2014) or counting directly the number of connected components (Gabrielov and Vorobjov, 2004).

Betti Numbers — Betti numbers are topological objects that can describe the complexity of topological spaces. More formally, the i-th Betti number of a space X is defined as the rank of the (finitely generated) i-th singular homology group of X. Roughly speaking, it counts the number of i-th dimensional holes of a space X [2] and captures a topological notion of complexity that can be used to compare subspaces of $\mathbb{R}^d$. The reader can refer to algebraic topology textbooks for a more comprehensive introduction to homology (Bredon, 2013; Hatcher, 2002). Informally, Betti numbers quantify the number of "holes" of various dimensions in a topological space. Each Betti number, b_i represents the number of i-dimensional 'independent' holes or cycles that cannot be continuously deformed into each other. For instance The zeroth Betti number, b_0, counts the number of connected components in the space. The first Betti number, b_1, counts the number of independent loops or one-dimensional holes. Higher Betti numbers, such as b_2, b_3, and so on, count holes of increasing dimensionality. To give some examples, 1. Circle (1-dimensional): A circle has one connected component ($b_0 = 1$) and one dimension hole or circle ($b_1 = 1$).

On the contrary, the unit sphere $\{x \in \mathbb{R}^3 : ||x||_2 = 1\}$, has one connected component ($b_0 = 1$) and no holes ($b_1 = 0$) but has one two-dimensional cavity ($b_2 = 1$) . A torus, like a doughnut, has one connected component ($b_0 = 1$) and two independent holes ($b_1 = 2$) and one two-dimensional cavity ($b_2 = 1$).

This distinction can be understood visually: a sphere is a solid shape with no internal holes, while a torus has a hole in its center and an additional loop around the hole.

When applied to the context of a loss function, Betti numbers can offer insights into the complexity of its optimization landscape, such as the presence of multiple local minima and regions of attraction. Betti numbers provide a tool to analyze the configuration of critical points and distinct regions in the optimization landscape

The following result connects the theory of Pfaffian functions and Betti numbers; in particular, it gives a bound on the Betti numbers for varieties defined by equations, including Pfaffian functions.

Theorem 5.2.5 (Sum of the Betti numbers for a Pfaffian variety (Zell, 1999))**.** *Let S be a compact semi-Pfaffian variety in $U \subset \mathbb{R}^{\tilde{n}}$, given on a compact Pfaffian variety V, of dimension n', defined by s sign conditions of Pfaffian functions.*

$$B(S) \in s^{n'} 2^{(\ell(\ell-1))/2} O((\tilde{n}\beta + \min(\tilde{n}, \ell)\alpha)^{\tilde{n}+\ell}). \tag{5.1}$$

In this work, the theorem is applied on the Pfaffian variety $S_{\mathcal{N}} = \{\theta \in U, \text{s.t. } \mathcal{L}(\theta) \leq c\}$, determined by the unique sign condition $\mathcal{L}(\theta) \leq c$, for a threshold c. This way, we will obtain a bound on the sum of the Betti numbers of $S_{\mathcal{N}}$. Therefore, we must demonstrate that the loss function is a Pfaffian function and compute its format. This goal can be achieved by writing the loss derivates regarding the network parameters and a Pfaffian chain.

In the following, we will use the chain rule of Backpropagation for such a purpose.

The constraints on the compactness of $\mathcal{U}$ and $\mathcal{V}$ can be removed without affecting the bounds as shown in Zell (2003). We will consider $\mathcal{U} = \mathbb{R}^n$ as the parameters' domain. Moreover, given that the semi-pfaffian variety is defined by one sign condition, $s = 1$.

[2]A i-th dimensional hole is a i-dimensional cycle that is not a boundary of a $(i + 1)$-dimensional manifold.

5.3 Main results

This section presents our theoretical analysis of the loss landscape topology. We start proving that, given a Pfaffian activation function σ of format $(\alpha_\sigma, \beta_\sigma, \ell_\sigma)$, the MSE loss function and BCE loss function computed over feedforward neural networks are also Pfaffian functions; their format is provided with an explicit dependency on the format of σ, on the number of layers and the number of neurons per layer.

Theorem 5.3.1 (MSE Loss). *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a function for which exists a Pfaffian chain $(\sigma_1, \dots, \sigma_\ell)$ and $\ell_\sigma + 1$ polynomials Q and P_i, $1 \leq i \leq \ell_\sigma$ of degree β_σ and α_σ , respectively s.t. σ is Pfaffian with format $(\alpha_\sigma, \beta_\sigma, \ell_\sigma)$.*

Moreover, let $g : \mathbb{R} \to \mathbb{R}$ be a function for which there exists a Pfaffian chain $(g_1, \dots, g_{\ell_g})$ and $\ell_g + 1$ polynomials, Q_g and P_g^i, $1 \leq i \leq \ell_g$ of degree β_g and α_g, respectively, s.t. g is Pfaffian with format $(\alpha_g, \beta_g, \ell_g)$:

Let $f(\theta, x)$ be the function implemented by a neural network with parameters $\theta \in \mathbb{R}^{\tilde{n}}$, input $x \in \mathbb{R}^{n_0}$, L layers and activation function σ for all layers except the last. The last layer can either have an activation function g or be linear.

Then, the MSE Loss function is Pfaffian with format

$$\left((degree(\sigma') + 1)(L - 2) + degree(\sigma'),\ 2(\beta_\sigma + 1),\ m\ell_\sigma \sum_{k=1}^{L-1} n_k \right)$$

when the last layer is linear. Here,

$$degree(\sigma') = \begin{cases} \beta_\sigma + \alpha_\sigma - 1 & case\ 1 \\ \beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1) & case\ 2 \end{cases}$$

where case 1 refers to the case where $P_\sigma^i(a, \sigma_1(a), \dots, \sigma_{\ell_\sigma}(a))$ don't depend explicitly on a, namely occurrences of a appear that are not of the type $\sigma_i(a)$; case 2 refers to the case where they depend on it. Moreover, we assume that the polynomials $P_g^i(a, g_1(a), \dots, g_{\ell_g}(a))$ don't depend explicitly on a .

The format of the chain becomes

$$\left((degree(\sigma') + 1)(L - 2) + degree(\sigma') + degree(g') + 1, 2\beta_g, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g) \right).$$

if the non-linearity g is applied also to the last layer.

For the BCE Loss function, we only explore the case where the last layer contains a non-linearity, namely $f_\theta(x) = g(a^L)$ since BCE loss is commonly used in binary classification problems where the output is a probability of the input belonging to one of two classes. In such problems, the last layer of the model typically uses a sigmoidal activation function to ensure that the output is between 0 and 1, representing the probability of the input being in class 1.

Theorem 5.3.2 (BCE Loss). *Let the hypothesis of Theorem 5.3.1 hold. If the activation function g is also used to the last layer, the BCE Loss function is Pfaffian with the format*

$$\left((L - 2)(degree(\sigma') + 1) + degree(\sigma') + degree(g') + 3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g + 4) \right).$$

In case the last activation function is the sigmoid function, the BCE Loss function has a format

$$\left((L-2)(degree(\sigma')+1)+degree(\sigma')+3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1\right).$$

Theorems 5.3.1 and 5.3.2 hold in general for any Pfaffian activation function and any sequence layer widths $(n_0, n_1, \dots, n_L)$. In the following, we specialize the results to the case when the activation function is either a sigmoid or a hyperbolic tangent. Moreover, for the sake of simplicity, We assume that all the hidden layers have the same width h.

Corollary 5.3.2.1. *Let us consider a feedforward perceptron network where all hidden layers have the same width h. The activation function can be either the hyperbolic tangent (*tanh*) or the sigmoid function (*logsig*), and the loss function is the Mean Squared Error. The last activation is assumed to be the sigmoid function. In this setting, the following holds*

- *when the last layer of the network is linear, the Pfaffian format of the loss function is given by*
$$(3(L-2),\ 4,\ m\,(L-1)\,h)\,;$$
- *when the last layer is non-linear, the Pfaffian format of the loss function is*
$$(3(L-2)+5,\ 2,\quad m(h(L-1)+1))$$

We can state an analogous result for the BCE loss function.

Corollary 5.3.2.2. *For a feedforward perceptron network with all the hidden layers have the same width h and trained using BCE loss function and a non-linear last layer, the following hold:*

- *if the non-linearity used is the sigmoid function, the Pfaffian format of the loss function is*
$$(3(L-2)+5,\ 1,\quad m\,((L-1)h+1)+1)$$
- *if the non-linearity used is* $\tanh(x)$*, the Pfaffian format of the loss function is*
$$(3(L-2)+7,\ 1,\quad m((L-1)h+5)).$$

We now state the main results of our work.

The presented theorem investigates the dependence of the sum of Betti numbers associated with the semi-Pfaffian variety $S_\mathcal{N}$, which represents the parameter set where the loss is non-negative, on factors such as the number of samples, network width, and network depth. Using Corollaries 5.3.2.1 and 5.3.2.2 we can derive the bounds on the Betti numbers of the Pfaffian semi-variety for both MSE and BCE loss function.

Theorem 5.3.3. *Let us consider a deep feedforward perceptron network $\mathcal{N}$ with $L \geq 3$ layers of width h. The activation function can be either the hyperbolic tangent or the sigmoid function; the loss is either the MSE or the BCE function, and the last layer is either linear (only for MSE) or nonlinear. Moreover, let us denote by S the semi-Pfaffian variety given by the set of parameters where the loss function is non- negative,*

i.e. $S = S_\mathcal{N} = \{\theta | \mathcal{L}(\theta) \geq c\}$ for a threshold $c \in \mathbb{R}^+$. Then, and by $B(S)$ the sum of the Betti numbers $B(S)$ of is bounded as follows.

- *With respect to the number of samples m, fixing h and L as constants,/ we have that $B(S) \in r^{O(m^2)}$, where r is a constant greater than 2.*
- *With respect to h, we have $B(S) \in O(h^2)^{O(h^2)}$.*
- *Finally, with respect to L, $B(S) \in 2^{O(L^2)} O(L^2)^{O(L)}$ holds.*

On the other hand, in the case of a shallow network with one hidden layer, i.e., $L = 2$, the following results hold.

- *With respect to m the bound is the same we obtained for deep networks, $B(S) \in c^{O(m^2)}$.*
- *With respect to h, we have $B(S) \in O(h)^{O(h)}$.*

It is worth emphasizing that the bounds concerning h and L offer insights into the relationship between the topological complexity of the loss landscape and the total number of parameters $\tilde{n}$. Specifically, in both cases, as $\tilde{n}$ varies, we explore the effect of changing the network width by treating h as a variable while keeping L fixed. Conversely, by treating L as a variable while keeping the width fixed, we investigate the impact of altering the network depth.

A first major remark from Theorem 5.3.3 is that the upper bound on the Betti numbers associated to the loss function is only exponential in the number of samples m, while it is superexponential in the number of neurons h or in the number of layers L. Intuitively, the take-home message is that the topological complexity of the loss function is less conditioned by the number of samples than by the number of parameters.

As it may not appear surprising, Theorem 5.3.3 also suggests that the complexity of the loss landscape with respect to deep networks increases with the number of neurons h at a much faster pace than the one with respect to the shallow networks, going from a dependence of the type $O(h^2)^{O(h^2)}$ to a dependence of the type $O(h)^{O(h)}$. Such a difference in behavior is coherent with results present in literature (Li et al., 2018), where it is proven that, when networks become sufficiently deep, neural loss landscapes quickly transition from being nearly convex to highly chaotic.

5.4 Complete proofs

5.4.1 Making derivatives explicit using backpropagation

Let $\mathcal{L}(\theta) = \frac{1}{m}\sum_{i=1}^{m} loss(f_\theta(x_i), y_i)$ be a generic loss. We aim to determine the gradient for a given input-output pair (x_i, y_i), with respect to the weight variables w^l_{jk} (connecting the j-th neuron of layer $l-1$, with the k-th neuron of layer l), which are elements of the augmented weight matrix W^l. The gradient components $\frac{\partial \mathcal{L}}{\partial w^l_{jk}}$ can be calculated through the chain rule. The Backpropagation algorithm provides an efficient method of spreading the error contribution back through the layers for updating weights.

Let us define $\delta^l_k = \frac{\partial \mathcal{L}}{\partial a^l_k}$, for $l = 1, \ldots, L$, as the derivative of the cost function with respect to the activation a^l_k of the k-th neuron of layer l. Then

$$\frac{\partial \mathcal{L}}{\partial w^l_{i,j}} = \delta^l_j z^{l-1}_i,$$

which represents a polynomial function in δ_j^l, z_i^{l-1}, so that all δ_j^l, z_i^{l-1} and their derivatives belong to the Pfaffian chain describing $\mathcal{L}$. Moreover, by the chain rule we have that

$$\delta_j^l = \frac{\partial \mathcal{L}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \frac{\partial \mathcal{L}}{\partial a_k^{l+1}} \frac{\partial a_k^{l+1}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \delta_k^{l+1} \frac{\partial a_k^{l+1}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \delta_k^{l+1}\, w_{k,j}^{l+1} \sigma'(a_j^l),$$

which means that δ_j^l is polynomial in all n_{l+1}, δ_i^{l+1} and $\sigma'(a_j^l)$, so that we have also to include all δ_j^{l+1} and all $\sigma'(a_j^l)$ and their derivatives in the Pfaffian chain.

Summing up, fixing an input x_i, y_i and proceeding backward through the layers, we can derive that

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \operatorname{poly}(&\delta_1^L, \dots, \delta_{n_L}^L, \sigma'(a_1^L), \dots, \sigma'(a_{n_L}^L), \dots, \\ &\sigma'(a_1^{l+1}), \dots, \sigma'(a_{n_{l+1}}^{l+1}), \sigma'(a_j^l), \sigma(a_i^{l-1})). \end{aligned} \tag{5.2}$$

Remark 6. *Notice that if σ is Pfaffian. It follows that the derivative σ' is polynomial in the factor of the chain, and the degree of the polynomial is at most α_σ, while the degree of σ in its chain is β_σ.*

Consequently, being $\frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \delta_j^l z_i^{l-1}$, we have:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = poly(&\delta_1^L, \dots, \delta_{n_L}^L, \sigma_1(a_1^L), \dots, \sigma_{\ell_\sigma}(a_1^L), \dots, \sigma_1(a_{n_L}^L), \\ &\dots, \sigma_{\ell_\sigma}(a_{n_L}^L), \dots, \sigma_1(a_1^{l+1}), \dots, \sigma_{\ell_\sigma}(a_1^{l+1}), \dots, \sigma_1(a_{n_{l+1}}^{l+1}), \\ &\dots, \sigma_{\ell_\sigma}(a_{n_{l+1}}^{l+1}), \dots, \sigma_1(a_j^l), \dots, \sigma_{\ell_\sigma}(a_j^l), \sigma_1(a_i^{l-1}), \dots, \sigma_{\ell_\sigma}(a_i^{l-1})). \end{aligned} \tag{5.3}$$

5.4.2 Proof of Theorems 5.3.1 and 5.3.2

Preliminaries

We want to prove that the MSE loss function and the BCE loss functions are Pfaffian functions with respect to the parameters of the network, in the hypothesis that the non-linearities σ and g are Pfaffian. To do so, we need to find a Pfaffian chain so that the loss function can be written as a polynomial in that chain, and we need to compute the degree of this polynomial in the parameters and the maximum degree of the derivatives of the functions in the chain with respect to the parameters of the network.

Notice that in this particular case of the MSE,

$$loss_{\text{MSE}}(f(x_i), y_i) = \frac{1}{2}(f(\theta, x_i) - y_i)^2.$$

meaning that if f is a Pfaffian function of a given format $(\alpha_f, \beta_f, \ell_f)$, the loss is a Pfaffian function with respect to the same chain with format $(\alpha_f, 2\beta_f, \ell_f)$.

In the case of the BCE loss function,

$$loss_{\text{BCE}}(f(x_i), y_i) = -y_i \log(f_\theta(x_i)) - (1 - y_i) \log(1 - f_\theta(x_i)).$$

always assuming that f is a Pfaffian function of a given format $(\text{degree}(f') - 1, \beta_f, \ell_f)$ we can consider two possible chains. The first one is the chain in which we add to the chain of f the functions $\log(f_\theta(x_i))$ and $\log(1 - f_\theta(x_i))$ and their derivatives meaning that the format of the chain becomes $(\max\{\text{degree}(f') + 2, \alpha_f\}, 1, \ell_f + 4)$, where degree$(f')$ is the degree of f' as polynomial in the chain, we will specify which chain in the various cases. In case we have a sigmoid as the final activation function, we could consider a different chain in which we include the loss in the chain; in this case, to obtain a pfaffian chain we also need to include the function the sigmoid of f, $\sigma(f(x))$ so the format becomes $(\text{degree}(f') + 2, 1, \ell_f + 2)$.

To determine the degree of f' we will need to compute the degree of σ', this will be useful for all the different cases considered, so we're doing it in this section.

If $y_i = \sigma_i(a)$:

$$\begin{aligned}\frac{d\sigma(a)}{da}\bigg|_{a=a_h^l} &= \frac{dQ(y_1,\dots,y_\ell)}{da}\bigg|_{a=a_h^l} \\ &= \Big(\sum_{s=1}^{\ell} \frac{\partial Q(y_1,\dots,y_\ell)}{\partial y_s} P_u(a, y_1, \dots, y_u)\Big)\bigg|_{\substack{1\le u\le \ell \\ y_u=\sigma_u(a_h^l) \\ a=a_h^l}}\end{aligned}$$

$\frac{\partial Q(y_1,\dots,y_\ell)}{\partial y_s}$ has degree $\beta_\sigma - 1$ and $P_u(a, y_1, \dots, y_u)$ has degree α_σ in $\sigma_1(a), \dots, \sigma_\ell(a)$. In conclusion $\frac{d\sigma(a)}{da}\Big|_{a=a_h^l}$ is a polynomial of degree $\beta_\sigma + \alpha_\sigma - 1$ if, $\forall i$, P_i does not depend directly on a, and it is $\beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1)$ in the general case.

Indeed if P_i depends on a, we have that $P(a_h^l, \sigma_1(a_h^l), \dots, a_h^l)$ has degree $\alpha_\sigma(\beta_\sigma + 1)$, since the degree of a_h^l is $\beta_\sigma + 1$ in the Pfaffian chain. Notice that $a_h^l = W^l \sigma(a^{l-1})$ and $\sigma(a^{l-1})$ has degree β_σ.

Summarizing :

$$\text{degree}(\sigma') = \begin{cases} \beta_\sigma + \alpha_\sigma - 1 \ \text{ case 1} \\ \beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1) \text{case 2} \end{cases} \tag{5.4}$$

Where case 1 refers to the case where $P_\sigma^i(a, \sigma_1(a), \dots, \sigma_{\ell_\sigma}(a))$ don't depend explicitly a and case 2 where they depend on it.

MSE loss function

In our hypothesis $n_L = 1$, so δ^L, a^L and $f_\theta(x)$ are scalars. Depending on whether the last layer is linear or the non-linearity g is applied we have that

$$f_\theta(x) = \begin{cases} a^L \\ g(a^L). \end{cases}$$

Linear last layer In the case of linear activation, given that $a^L = W^L \sigma(a^{L-1})$ and that σ is Pfaffian with respect to the chain $\sigma_1, \dots, \sigma_{\ell_\sigma}$ we obtain that $f_\theta(x) = a^L$ is polynomial in the functions following chain

$$(((\sigma_k(a_i^j))_{k=1,\dots\ell_\sigma})_{i=1,\dots,n_j})_{j=1,\dots,L-1}$$

The degree of the Pfaffian function f in this chain is $\beta_f = \beta_\sigma + 1$; the maximum degree of the derivatives depends on the degree of σ' in (5.4) and is given by the chain rule, the worst case is given by deriving of the term of the vector $\sigma(a^{L-1})$ with respect to one of the weights of the first layer. In this case, applying thee chain rule and going backward layer by layer, we obtain that every step, we multiply the weight of one layer and the σ' it follows that the degree of $\frac{\partial \sigma(a^{L-1})}{\partial w^1_{i,j}}$ is $(\text{degree}(\sigma')+1)(L-2)+\text{degree}(\sigma')$

Taken in account what said in Section 5.4.2, we obtain that for a single input point, x, we obtained that the MSE loss with linear activation for the last layer is Pfaffian with respect to the following chain of length $\ell_\sigma \sum_{k=1}^{L-1} n_k$:

$$(((\sigma_k(a_i^j))_{k=1,\dots\ell_\sigma})_{i=1,\dots,n_j})_{j=1,\dots,L-1} \tag{5.5}$$

The order of the chain is given, going from the inner cycle to the outer cycle

Considering that we have to consider all the input points, the length of the chain becomes $m\ell_\sigma \sum_{k=1}^{L-1} n_k$. The format of the chain of the MSE loss function is therefore

$$((\text{degree}(\sigma')+1)(L-2)+\text{degree}(\sigma'),\ 2(\beta_\sigma+1),\ m\ell_\sigma \sum_{k=1}^{L-1} n_k) \tag{5.6}$$

Non linear last layer with non linearity g In this case, we need to add to the chain described in Equation (5.5) the terms $(g_k(a^L))_{k=1,\dots,\ell_g}$. The final chain will be:

$$[(((\sigma_k(a_i^j))_{k=1,\dots\ell_\sigma})_{i=1,\dots,n_j})_{j=1,\dots,L-1}, (g_k(a^L))_{k=1,\dots,\ell_g}] \tag{5.7}$$

The degree of the function f in this chain is given by β_g, the maximum degree of the derivatives is the degree of $\frac{\partial g(a^L)}{\partial w^1_{i,j}}$ and is $(\text{degree}(\sigma')+1)(L-2)+\text{degree}(\sigma')+\text{degree}(g')+1$.

Using the argument we used before for σ', we have that the degree of $g'(a)$ is $\beta_g+\alpha_g-1$ if, $\forall i$, P_g^i does not depend directly on a, and it is $\beta_g+\alpha_g-1+\alpha_g(\beta_g+1)$ in the general case.

$$\text{degree}(g') = \begin{cases} \beta_g+\alpha_g-1 & \text{case 1} \\ \beta_g+\alpha_g-1+\alpha_g(\beta_g+1) & \text{case 2} \end{cases} \tag{5.8}$$

Summarizing the format of the chain in the case of non-linear activation for the last layer is

$$((\text{degree}(\sigma')+1)(L-2)+\text{degree}(\sigma')+\text{degree}(g')+1, 2\beta_g, m(\ell_\sigma \sum_{k=1}^{L-1} n_k+\ell_g)) \tag{5.9}$$

5.4.3 BCE loss function

Non-linear activation g different from the sigmoid function In this case, the Pfaffian chain for the loss function will be the chain described in 5.7 to which we add the following terms $\frac{1}{g(a^L)}, \log(g(a^L)), \frac{1}{1-g(a^L)}, \log(1-g(a^L))$. The length of the chain will be $\sum_{k=1}^{L-1} n_k+\ell_g+4$. The degree of the loss function with respect to this chain is 1 and the maximum degree of the derivatives is given by the degree of the following derivative $\frac{\partial 1/g(a^L)}{\partial w^1_{i,j}}$ that is $(L-2)(\text{degree}\sigma'+1)+\text{degree}(\sigma')+\text{degree}(g')+3$.

It follows that the format of the chain for the BC loss function with sigmoid activation for the last layer is

$$((L-2)(\text{degree}(\sigma')+1)+\text{degree}(\sigma')+\text{degree}(g')+3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g + 4))$$

Non-linear activation g is the sigmoid function In this case, we want to include the loss in the chain and use the trick of backpropagation introduced in section 5.4.2 to be sure that its derivatives are polynomial in the chain. Eq 5.2 shows that the derivative of the loss with respect to the parameters is poly in δ_i^L and in the terms of the chain (5.7), in this case with g equal to the sigmoid function.

If we consider only a single sample x and the output of the network $f_\theta(x) = g(a^L)$. We have that

$$\begin{aligned}\frac{\partial loss_{\text{BCE}}(y, f_\theta(x))}{\partial a^L} &= g'(a^L)\frac{1}{g(a^L)(1-g(a^L))}(y_i - f_\theta(x)) \\ &= \sum_{i=1}^{m} \frac{g'(a^L)}{g(a^L)(1-g(a^L))}(y_i - g(a^L)).\end{aligned} \tag{5.10}$$

If the non-linearity g is the sigmoid function $\left(g(x) = \frac{1}{1+e^{-x}}\right)$, the term $\frac{g'(a^L)}{g(a^L)(1-g(a^L))}$ becomes 1 and we don't need to deal with it and the degree of δ^L is the degree of $g(a^L)$, that is β_g that for the sigmoid function is 1. We recall indeed that the sigmoid function is Pfaffian with format $(2, 1, 1)$.

It follows that the format of the chain for the BC loss function with sigmoid activation for the last layer is

$$((L-2)(\text{degree}(\sigma')+1)+\text{degree}(\sigma')+\text{degree}(g')+1, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1)$$

The last $+1$ in the length is given by the fact that we're also adding the loss function computed on the input dataset to the chain. Moreover, we can compute degree(g') that is this case is 2, notice that this is smaller than the worst case proposed in (5.8) that would be equal to 4.

The final format of the chain will be

$$((L-2)(\text{degree}(\sigma')+1)+\text{degree}(\sigma')+3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1)$$

5.4.4 Proof of Corollary 5.3.2.1

Proof. It is enough to remark that the format of the hyperbolic tangent and the sigmoid function is $(\alpha_\sigma, \beta_\sigma, \ell_\sigma) = (2, 1, 1)$. Substituting these values in Theorem 5.3.1 leads straightforwardly to the statement.

□

5.4.5 Proof of Corollary 5.3.2.2

Proof. The result for the hyperbolic tangent activation function can be obtained by applying Theorem 5.3.2 with its corresponding format of $(2, 1, 1)$. On the other hand, for the sigmoid function, we have that $\sigma'(x) = \sigma(x)(1-\sigma(x))$. This allows us to obtain a linear dependency on $\sigma(x)$ in the derivative of the BCE loss function. Indeed,

$$\begin{aligned}
\frac{\partial \mathcal{L}(f)}{\partial \theta} &= -\frac{\partial}{\partial \theta}(y \log(\sigma(f(\theta))) + (1-y)\log(1-\sigma(f(\theta)))) \\
&= -[\frac{y}{\sigma(f(\theta))}\sigma'(f(\theta))\frac{\partial f(\theta)}{\partial \theta} - \frac{1-y}{1-\sigma(f(\theta))}\sigma'(f(\theta))\frac{\partial f(\theta)}{\partial \theta}] \\
&= -y(1-\sigma(f(\theta)))\frac{\partial f(\theta)}{\partial \theta} + (1-y)\sigma(f(\theta))\frac{\partial f(\theta)}{\partial \theta} \\
&= (\sigma(f(\theta)) - y)\frac{\partial f(\theta)}{\partial \theta}
\end{aligned}$$

□

5.4.6 Proof of Theorem 5.3.3

Proof. We can use Theorem 5.2.5 with $U = \mathbb{R}^{\tilde{n}}$, with $\tilde{n} = h(n_0+1) + h(h+1)(L-2) + h + 1 = h^2(L-2) + h(n_0+L) + 1$ the total number of parameters of the network.

In this case the term $s^{n'}$ in Equation (5.1) can be ignored, since $s = 1$.

Bounds for MSE loss function: deep case For $L \geq 3$:

- if the last layer has a linear activation, we have that

$$B(S) \in 2^{[m(L-1)h(m(L-1)h-1)]/2} O(f(n_0,h,L,m)^{h^2(L-2)+h(L+n_0+m(L-1))+1}) \tag{5.11}$$

  with $f(n_0,h,L,m) = 4[h^2(L-2) + h(L+n_0) + 1] + 3(L-2)\cdot\min(h^2(L-2) + h(L+n_0) + 1, m(L-1)h)$

  If $m >> h$ it becomes

$$\begin{aligned}
B(S) \in\, & 2^{(m(L-1)h(m(L-1)h-1))/2} \times \\
& O\Bigg(4\Big[h^2(L-2) + h(L+n_0) + 1\Big] \\
& + 3(L-2)\Big[h^2(L-2) + h(L+n_0) + 1\Big]\Bigg)^{h^2(L-2)+h(L+n_0+m(L-1))+1}
\end{aligned}$$

  If $h >> m$, and we consider L and m as constant it becomes

$$\begin{aligned}
B(S) \in\, & 2^{(m(L-1)h(m(L-1)h))/2} \times \\
& O\left(4[h^2(L-2) + h(L+n_0) + 1]\right. \\
& \left.\times 3(L-2)hm(L-1)\right)^{h^2(L-2)+h(L+n_0+m(L-1))+1}.
\end{aligned}$$

  It is important to note that the overparametrized regime falls within this scenario.

- if the last layer has a nonlinear activation we have that:

$$
\begin{aligned}
B(S) \in\, & 2^{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]/2} \\
& \times O\left(g(n_0, h, L, m)\right)^{h^2(L-2)+h(L+n_0+m(L-1))+2}
\end{aligned} \tag{5.12}
$$

  with $g(n_0, h, L, m) = 2[h^2(L-2) + h(L+n_0) + 1] + (3(L-2)+5) \cdot \min(h^2(L-2) + h(L+n_0) + 1, m(h(L-1)+1)$

  If $m >> h$ it becomes

$$
\begin{aligned}
B(S) \in\, & 2^{\frac{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]}{2}} \times \\
& O\Bigg(2\big[h^2(L-2) + h(L+n_0) + 1\big] \\
& + (3(L-2)+5)\Big(h^2(L-2) + h(L+n_0) + 1\Big)\Bigg)^{h^2(L-2)+h(L+n_0+m(L-1))+2}
\end{aligned}
$$

  If $h >> m$, and we consider L and m as constant it becomes

$$
\begin{aligned}
B(S) \in\, & 2^{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]/2} \\
& \times O\Big(2\big[h^2(L-2) + h(L+n_0) + 1\big] \\
& + (3(L-2)+5)m(h(L-1)+1)\Big)^{h^2(L-2)+h(L+n_0+m(L-1))+2}
\end{aligned}
$$

In both cases we can see that, as a function of the number of samples m, fixing h and L as constants, we have that $B(S) \in c^{O(m^2)}$, where c is a constant greater than 2. As a function of h, considering the others variables as constants $B(S) \in O(h^2)^{O(h^2)}$. Eventually, as a function of L, considering m and h as constants, $B(S) \in 2^{O(L^2)} O(L^2)^{O(L)}$.

Bounds for MSE loss function: shallow case In the case in which $L = 2$, all the terms in which $L-2$ occurs vanish. Therefore,

- if the last layer has a linear activation, equation 5.11 simplifies in the following way:
$$B(S) \in 2^{[mh(mh-1)]/2} O([4h(2+n_0)+4]^{h(2+n_0+m)+1})$$

- if the last layer has a nonlinear activation, since the number of samples is usually larger that the input dimension, $\min(h(2+n_0)+1, 2mh) = h(2+n_0)+1$. This simplify equation 5.12 in the following way:

$$B(S) \in 2^{[2mh(2mh-1)]/2} O([9h(2+n_0)+9]^{h(2+n_0+2m)+1})$$

As a function of m the results is the some we obtained for deep networks, $B(S) \in c^{O(m^2)}$. Conversely, as a function of h, the dependency change and we obtain $B(S) \in O(h)^{O(h)}$.

Bounds for BCE loss function: deep case For $L \geq 3$:

- if the last layer has a a sigmoid activation function we have that

$$\begin{aligned} B(S) \in & 2^{[(m((L-1)h+1)+1)(m((L-1)h+1))]/2} \\ & \times O\left(g(n_0,h,m,L)^{h(m(L-1)+n_0+2+(h+1)(L-2))+m+2}\right) \end{aligned} \tag{5.13}$$

  with $g(n_0,h,m,L) = h^2(L-2) + h(n_0+L) + 1 + [3(L-2)+5]\min(h^2(L-2) + h(n_0+L)+1, [m((L-1)h+1)+1])$
  If $m >> h$ $g(n_0,h,m,L)$ becomes

$$g(n_0,h,m,L) = h^2(L-2)+h(n_0+L)+1+[3(L-2)+5](h^2(L-2)+h(n_0+L)+1)$$

  On the other side, if $h >> m$ and we consider m and L as constant, we have that

$$g(n_0,h,m,L) = h^2(L-2)+h(n_0+L)+1+[3(L-2)+5][m((L-1)h+1)+1].$$

Bounds for BCE loss function: shallow case For $L = 2$:

- if the last layer has a non-linear activation function we have that

$$B(S) \in 2^{[(m(h+1)+1)(m(h+1))]/2} O(g(n_0,h,m)^{h(m+n_0+2)+m+2}) \tag{5.14}$$

  with $g(n_0,h,m) = h(n_0+2)+1+5\min(h(n_0+2)+1, m(h+1)+1)$
  If $m >> h$ $g(n_0,h,m,L)$ becomes

$$g(n_0,h,m) = h(n_0+2)+1+5(h(n_0+2)+1).$$

  On the other side, if $h >> m$ and we consider m and L as constant, we have that

$$g(n_0,h,m,L) = h(n_0+2)+1+5[m(h+1)+1].$$

Resulting in asymptotic bounds equal to those derived previously for the MSE loss with non-linear activation. The same holds with similar computations in case the last activation function is the hyperbolic tangent.

We remark that for the BCE loss, our focus is primarily on studying the case where the last layer of the neural network has a non-linear activation function since it is commonly used for binary classification tasks.

□

5.4.7 Residual connections

Without loss of generality, we can consider the case in which we utilize skip connections that provide the previous layer's output, through summation, as an additional input to the subsequent layer. In this case, denoting by z_i the output of the i-th layer, if we consider the case, we have that

$$\begin{aligned}
&z_1 = \sigma(W^1 x) \\
&z_2 = \sigma(W^2 z_1) + z_1 \\
&z_3 = \sigma(W^3 z_2) + z_2 \\
&\vdots \\
&z_l = \sigma(W^l z_{l-1}) + z_{l-1} \\
&\vdots
\end{aligned}$$

If we want to obtain the derivative of z_l with respect to a parameter of the network $w^k = W_i^k$ with $k < l$ and $i \in \{1, \ldots, n_k\}$, we have that

$$\frac{\partial z_l}{\partial w^k} = W^l \sigma'(W^l z_{l-1}) \frac{\partial z_l - 1}{\partial w^k} + \frac{\partial z_l - 1}{\partial w^k} \tag{5.15}$$

We observe that the derivative $\frac{\partial z_l - 1}{\partial w^k}$ appears twice in the equation. However, due to the multiplication with other terms in the first term, it only affects the degree of the polynomial in the first term of the sum. Therefore, we can disregard the second term in the equation when interested in computing the format of the Pfaffian chain. This reasoning can be extended to all layers, demonstrating that the degrees of the derivatives will remain unchanged, just as we computed for simple feedforward perceptron neural networks. Similarly, the length of the chain remains unaltered. In this case as well, the functions to be included in the chain are the outputs of the layers, which are exactly the same in number as those in feedforward networks, albeit with different forms.

5.4.8 Regularization terms and residual connections

The role of regularization

Remark 7. *(ℓ_2 regularization) One could be interested in seeing how our analysis is influenced with the introduction of regularization terms. We can face this new scenario in case we add an ℓ_2 regularization term, being the ℓ_2 regularization term $\Omega(\theta) = \sum_i \|\bar{W}_i\|^2$ polynomial in the parameters θ. This term only affects the term β of the format of the Pfaffian function $\bar{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \lambda\Omega(\theta)$, as it affects the degree of the polynomial with respect to the weights, adding a monomial term of degree 2 ; nevertheless, the bound on the Betti numbers is not affected by it. This can be easily derived from the computations in Section 5.4 for the proof of Theorem 5.3.3. The ℓ_2 regularization doesn't promote sparsity (which is instead promoted by the ℓ_1 regularization) (Hastie et al., 2009), but it affects the magnitude of the weights. This could suggest that the regularization term may just provide a scaling of the loss function, and therefore, all local minima may still be present; nevertheless, it is possible that our theoretical bounds may not be able to catch the influence of the regularization term in the loss landscape.*

Residual Connections

Skip connections or residual connections, are widely employed in neural networks to alleviate the vanishing gradient problem and enhance information flow across layers. his architectural design was popularized by the ResNet model (He et al., 2016) and has since been adopted in various network architectures.

Remark 8. *Introducing a residual term at each layer in a neural network, thus creating a Residual Neural Network (ResNet), does not impact the bounds provided in our analysis. It does not affect the number of functions required in the chain or the maximum degree of the polynomials. The addition of skip connections combines the output of a previous layer with that of a subsequent layer through summation. Regarding our analysis, the essential factors, such as the degree of polynomials and the length of the Pfaffian chain, remain the same. The only change lies in the specific terms included within the chain. See section 5.4.7 in Section 5.4 for more details. Consequently, it implies that utilizing a ResNet rather than a conventional feedforward neural network does not alter the topology of the loss function or its optimization process. Instead, the primary advantage lies in enhancing the network's expressive capacity.*

Skip connections allow gradients to flow more easily during backpropagation, facilitating the training of deeper networks. It has been observed that skip connections promote flat minimizers and prevent the transition to chaotic behavior (Li et al., 2018). Our current theoretical framework is limited in capturing the reduced complexity of the loss landscape induced by skip connections. Specifically, our theory provides an upper bound on the number of minima, lacking a lower bound, which implies that our estimate may exceed the actual value. Moreover, the analysis in Li et al. (2018) focuses solely on the minima, while our proposed bound encompasses the sum of all non-zero Betti numbers. Finally, it is worth observing that the optimisation algorithms do not explore the whole space but only a part where the complexity of the function might be lower. Therefore, regularisation and skip connections could be mechanisms for which only submanifolds are actually explored by the optimisation algorithm, and such a behaviour could not be captured by what the global bound suggests.

5.5 Conclusions

Our investigation determined that when employing a Pfaffian non-linearity, both the MSE and BCE loss functions can be represented as Pfaffian functions. Subsequently, we analyzed the respective Pfaffian chains obtained in each case. Specifically, we examined the differences in the complexity and performance of the Pfaffian chains resulting from the use of the two loss functions.

When studying the complexity of the loss landscape, a superexponential dependency on the network parameters has been found; interestingly, a qualitative difference can be highlighted between the shallow and the deep case as we focus on the impact of the number of neurons h. Indeed, as the number of layers increases, the superexponential dependency involves a term h^2, and not h anymore. This result is aligned with the general intuition and previous works in literature (Bianchini and Scarselli, 2014). In any case, the asymptotic analyses show that the sum of Betti numbers has exponential dependences on the square of the number of samples m.

It's worth underlying the characterization of the topological complexity we derived for loss functions with an additional ℓ_2 term; from our analysis point of view, it seems that the presence of a regularization term is not implied in the design of the loss landscape, pointing out to a different role of the regularization itself in the network training, e.g. the optimization process. Nevertheless, being those boundaries not tight, space for a deeper analysis is not left out.

Bounds provided by the sum of Betti numbers are not tight; our analysis suggests a qualitative interpretation more than a quantitative one. Obtaining a bound on the number of connected components $b_0(S)$ rather than $B(S)$ would give a more accurate characterization of the topology of the loss landscape; this is a perspective to be considered for future works.

Chapter 6

Conclusions

In this dissertation, we have investigated generalization and robustness in machine learning through diverse research works, providing valuable insights and advancements towards building more reliable systems.

In this final chapter of the thesis, we revisit the research questions from the first chapter, state our main findings and identify directions for future work.

6.1 Leveraging inter-rater agreement for classification in the presence of noisy labels

In Chapter 2 we aimed to answer the following questions:

RQ1 *How can inter-annotator statistics be leveraged in learning with noisy labels? Is there a better way to aggregate labels than majority vote?*

RQ2*Is it possible to learn from datasets with noisy labels while still having performance guarantees?*

6.1.1 Main findings

We answered the first question affirmatively by finding a way to exploit the inter-annotator statistics to estimate the noise rate in the dataset. We also show that using soft labels based on the posterior distribution of the class of a sample given all the annotations for that sample is more beneficial than majority vote aggregation. We also partially answer affirmatively to the second question, proving generalization bounds in the case of a specific noise-resistant loss function. In the following paragraphs, we show our findings in more detail.

Estimation of Noise Distribution. We can estimate the noise distribution by computing an inter-annotator agreement matrix that measures the agreement between two annotators on the dataset. The inter-annotator agreement matrix between two annotators is a $C \times C$ matrix, where C is the number of classes. Its formal definition is $(M_{ab})_{i,j} := \mathbb{P}(y_a = i, y_b = j)$ where a and b are the two annotators. From an empirical estimation of this matrix and of the distribution of the classes, considered to be known, we can compute an estimation of the noise transition matrices, which is also a matrix of the size $C \times C$. The noise transition matrix represents the probabilities of altering true labels within a dataset. It outlines the probability of a true label being changed to another. Specifically, the matrix entry T_{ij} describes the probability that a true label i is corrupted and observed as label j.

The approach we derived is valid in the case of instance-independent noise, where the probability of label corruption is independent of the input of the sample but depends only on its class. We proved the consistency of the estimator of the noise transition matrix and provide bounds on the estimation error.

Methods for Learning from Noisy Datasets. Once an estimator for the noise transition matrix is obtained, it can be exploited to define a robust loss function. In particular, leveraging the Bayes theorem from an estimation of the noise transition matrix, we derived the posterior distributions of the class of a sample given all the annotations for that sample. We also noticed that the posterior distribution for every sample calculated using the true noise transition matrix converges to the Dirac delta distribution centred on the true label almost surely. We proposed the usage of the posterior so computed as soft labels that can be used during training in the cross-entropy loss. The method utilizes the estimated noise distribution to improve learning from noisy datasets.

Generalization Bounds. We showed that alternatively, the estimated matrix can be used in two noise-robust loss functions introduced by Patrini et al. (2017), precisely the backward and forward loss. For some particular aggregation methods of the annotations provided by the multiple annotators, we obtained generalization bounds of the excess risk. The derived bounds rely on the Rademacher complexity of the function space and the Lipschitz constant of the loss function. These bounds are significant as they provide performance estimates for a model trained on noisy data, utilizing values estimable from the noisy dataset. Unlike other approaches, our method doesn't hinge on the true noise transition matrix of annotators, which is typically inaccessible from training data. More specifically, these bounds are contingent on the estimated noise transition matrix, the number of classes, the Rademacher complexity, and the Lipschitz constant, all of which can be assumed known beforehand. Additionally, the bounds consider the ground truth distribution, often assumed to be uniform in various scenarios.

Experimental Validation. We conducted experiments to validate the effectiveness of our proposed method for estimating the noise transition matrix by assessing estimation errors concerning the number of samples. Additionally, we explored the usage of the estimated posteriors as soft labels. The experiments encompassed a classification task on a synthetic dataset and CIFAR10-N dataset, a real dataset containing multiple noisy annotations. Our findings indicate that employing the posterior distribution as soft labels leads to improved performance compared to using average labels from annotators through majority or random aggregation methods. Our method demonstrated increased resilience to noise and displayed reduced variance in results. This supports our hypothesis that leveraging the estimation of the noise transition matrix contributes to enhanced classification accuracy.

6.1.2 Future directions

This work represents a significant step in addressing the challenge of learning from datasets with noisy labels due to not adversarial annotator errors. Looking ahead, the potential for future research lies in extending our approach beyond the constraints of symmetry for the noise transition matrix that we require in our theorems and the assumption of knowing the distribution of the classes in the dataset, aiming to explore scenarios where noise transition matrix may not be symmetric and may differ among annotators. Furthermore, our next step is to extend the utility of this method across a spectrum of distinct domains and applications. One possibility entails the domain of medical diagnostics, wherein annotations are sourced directly from medical professionals. In this context, applying our methodology can enhance diagnostic accuracy when multiple annotations can be obtained. Additionally, another promising domain involves ranking and learning to rank when each training instance is labeled by multiple annotators.

6.2 On generalization bounds for clustering

In Chapter 3 we investigated the following questions:

RQ3 *How does the excess risk behave as function of k, d, and n for center-based (k, z) clustering ?*

RQ4 *How does the excess risk behave as function of k, d, j, and n for subspace (k, j, z) clustering ?*

6.2.1 Main findings

We addressed the questions, presenting nearly optimal results. Specifically, we derived bounds for the excess risk for the various clustering settings. For the (k, z)-clustering, utilizing n independent samples from an unknown fixed distribution $\mathcal{D}$, we established the bound at $\tilde{O}\left(\sqrt{k/n}\right)$. This bound aligns precisely with the lower boundary as determined by Bartlett et al. (1998c).

Similarly, our exploration of (k, j, z)-clustering derived an excess risk bound at $\tilde{O}\left(\sqrt{kj^2/n}\right)$, using the same set of independent samples from the distribution.

Moreover, our analysis unveiled the existence of a distribution demonstrating an excess risk of at least $\Omega\left(\sqrt{kj/n}\right)$ for the $(k, j, 2)$-clustering problem. This finding closely corresponds to the upper boundary established by Fefferman et al. (2016), accounting for polylog factors.

In both settings, we pursued the same route to establish the upper bounds. These bounds are derived by constraining the excess risk through a uniform bound of the difference between the empirical cost and the cost related to the distribution for any provided solution. The uniform bound can be, in turn, bounded by the Gaussian complexity of the cost vectors. Employing the chaining techniques alongside the concepts of ε-net enabled us to estimate the Gaussian complexity of the cost vectors.

However, a dependence on the dimension where the points originally lived persisted. To overcome this in center-based clustering, we leaned on terminal embeddings. Conversely, for subspace clustering, terminal embeddings are an infeasible solution as they aren't linear and do not preserve subspaces. Consequently, we established the existence of a collection of dimension reducing maps so that each subspace is preserved by at least one embedding within the collection. Technically, this contribution stands as one of our primary focal points.

We also presented a lower bound specific to projective clustering, confirming the near-optimality of the results derived by Fefferman et al. (2016).

Experimental Validation. Moreover, we substantiate our theoretical bounds with empirical experiments conducted on real datasets. The empirical results affirm that the practical learning rates align closely with the theoretical guarantees, implying near-optimality even in real-world applications. This outcome defies the anticipation that practical experiments might deviate from the bounds established for worst-case scenarios. In particular, the distribution of examples could potentially have properties that allow for a better convergence rate. Moreover, we noticed that the rates were not particularly affected by either the choice of z or by the dimension j when analyzing subspace clustering.

6.2.2 Future directions

Examples of open problems that we would like to study concern clustering problems with clustering size constraints. For instance, one could consider a fair clustering problem where the clusters are asked to have a balanced composition between classes (Schmidt et al., 2019; Bandyapadhyay et al., 2020). In this case, what kind of rates can be obtained? Can we still utilize the same techniques? Notably, the constraints on cluster compositions for individual centers disrupt the independence in point assignments to these centers. This scenario necessitates alternative methodologies to address the interdependence of these assignments. Another open problem concerns obtaining better bounds when some favourable conditions for clustering hold for the points' distribution. For instance, conditions on the margin separating the clusters. In this case, can we improve the upper bounds? And under which exact conditions?

6.3 A new generalization of the artificial neuron to enhance the interpretability of neural networks while preserving expressive capabilities

Chapter 4 investigated the following question:

RQ5 *Is it possible to design a neural network structure that makes the whole model interpretable without sacrificing effectiveness and expressiveness?*

To the first question, we answered partially yes. Or better, in Chapter 4, we presented a new artificial neuron based on the inversion between the operation of the sum and the application of the non-linearity. To simplify, we could describe the structure of the new neuron as follows: each dimension of the input vector (a scalar number) is first shifted or translated by a factor b, the bias and then the shifted value is scaled by a factor w (the weight). Subsequently, a non-linearity is applied. Finally, an aggregation operation is applied to all the output of the input feature to aggregate them. Performing operations in this order makes it possible that when the Heaviside function is used as non-linearity, we can derive exact rules to determine the impact of the original features on the final prediction. When the non-linearity is a continuous function, such as a sigmoid or product of hyperbolic tangent, the rules we can derive are "fuzzy".

Universality. In terms of their ability to represent and process information, networks using this new artificial neuron possess the same expressive capacity as conventional neural networks. More specifically, when the processing function is the Heaviside function we proved that the network could approximate any continuous function on the unit hypercube, I_n, Lebesgue measurable functions on I_n and functions in $L^p(A, \mu)$ for $1 \leq p < \infty$, with μ being a Radon measure and $A \in \mathcal{B}(\mathbb{R}^n)$ a Borel set.

Performances. In conclusion, our experiments with synthetic and real data have shown that our method is better than popular interpretable by design methods like Decision Trees and Logistic Regression, and it can do as well as the regular types of neural networks and Gradient Boosted Decision Trees, which instead lack transparency in their operations.

Among the Heaviside, sigmoid, and tanh-prod cases, the Heaviside case yields the least favorable outcomes. This could be attributed to its greater complexity in training, despite being the most interpretable among the three. Conversely, the tanh-prod case displays marginally superior performance compared to the sigmoid, offering increased flexibility.

6.3.1 Future directions

In future work, we'll look more closely at how the new neuron-based models work when we change the number of layers, the number of neurons per layer, and the processing functions used. Also, we will check how different initializations of these models can affect the performances.

Moreover, we'll try to generalize this structure to other types of networks, like Recurrent, Convolutional and Graph Neural Networks.

6.4 A topological description of loss surfaces via Betti numbers characterization

Chapter 5 considered the following questions:

RQ6 *Can a topological measure effectively assess the complexity of the loss implemented by layered neural networks?*

RQ7 *How do the complexity bounds of deep and shallow neural architectures relate to the number of hidden units and the selected activation function?*

Our investigation determined that when employing a Pfaffian function as activation function of the neural network, both the MSE and BCE loss functions can be represented as Pfaffian functions. Subsequently, the theory developed for Pfaffian functions in Zell (2003, 1999) allowed us to obtain a bound of the sum of Betti numbers of the sublevel set of the empirical risk of the loss. The sum of Betti numbers, in general, can be seen as a measure of the topological complexity of a space. The Betti numbers characterize the shape or structure of topological spaces. The sum of these numbers, taken across various dimensions, offers a comprehensive view of the overall topological complexity. A higher sum of Betti numbers indicates a more complex and rich topological structure, reflecting more intricate features like a higher number of connected components, holes, or higher-dimensional cavities.

Specifically, we examined the differences in the complexity of the Pfaffian function resulting from using the two loss functions.

We noticed that the bounds in this work on the sum of Betti numbers of the sublevel set of the empirical risk are not tight; our analysis suggested a qualitative interpretation more than a quantitative one. The ideal scenario would be obtaining a bound on the number of connected components of the sublevel set $b_0(S)$ rather than $B(S)$, which would give a more accurate characterization of the topology of the loss landscape for the purpose of gradient descent based algorithms; this is a perspective to be considered for future works.

When we look at how complex the loss landscape is, we obtain a superexponential dependency of the bound on the network parameters. Particularly, a qualitative difference can be highlighted between the shallow and the deep case as we focus on the impact of the number of hidden neurons h per layer. For shallow networks, the superexponential dependency involves a term h, while for deep networks, it now involves terms of the type h^2. This result is aligned with the general intuition and previous works in literature (Bianchini and Scarselli, 2014).

Both for deep and shallow networks and all loss fucntions, the asymptotic analyses showed that the sum of Betti numbers of the sublevel set of the empirical loss has exponential dependences on the square of the number of samples m.

6.4.1 Future directions

Future research can focus on refining the characterization of the loss landscape's topology by obtaining more precise bounds on the number of connected components of the sublevel set or the level set of the empirical risk. This finer characterization could significantly enhance the understanding of the topological properties for optimizing gradient-descent-based algorithms.

Closing Remarks

In conclusion, this thesis aimed to obtain more reliable machine learning models.

Initially, we focused on enhancing the reliability of models when trained on noisy data. Real-world data often comes with imperfect labeling, driving our focus to create robust learning algorithms for noisy labels. In particular, the method we introduced is suitable in scenarios where multiple labels are provided by different annotators.

Furthermore, we focused on obtaining nearly optimal bounds for the excess risk of clustering objectives, driven by the need to improve how we can rely on the chosen model when the model is asked to perform on data not seen during training. The bounds obtained are upper bounds for the rate of decreasing of the excess risk in terms of the number of samples and the number of centers.

Moreover, we introduced a novel artificial neuron designed to enhance model interpretability. Our research ensured that networks built with these neurons are not only more interpretable but also preserve the same expressive power as those utilizing standard neurons. Indeed, we proved that such networks can approximate the same sets of functions as standard neural networks and verified their comparable performance to standard neural networks on the datasets we tested.

Finally, we focused on a topological analysis of loss functions. The goal of providing critical insights into the structure of loss landscapes stems from the fact that the study of loss landscapes is of interest to the behavior of the gradient-based algorithm. Consequently, studying loss landscape is interesting because it affects the convergence and stability of the optimization method and significantly influences the reliability of a model.

Bibliography

Abaya, E. F. and Wise, G. L. (1984). Convergence of vector quantizers with applications to optimal quantization. *SIAM Journal on Applied Mathematics*, 44(1):183–189.

Agarwal, P. K., Har-Peled, S., and Varadarajan, K. R. (2004). Approximating extent measures of points. *J. ACM*, 51(4):606–635.

Angelidakis, H., Makarychev, K., and Makarychev, Y. (2017). Algorithms for stable and perturbation-resilient problems. In Hatami, H., McKenzie, P., and King, V., editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 438–451. ACM.

Anthony, M. and Bartlett, P. L. (1999). *Neural Network Learning: Theoretical Foundations*. Cambridge University Press.

Antos, A., Gyorfi, L., and Gyorgy, A. (2005). Individual convergence rates in empirical vector quantizer design. *IEEE Transactions on Information Theory*, 51(11):4013–4022.

Apicella, A., Donnarumma, F., Isgrò, F., and Prevete, R. (2021). A survey on modern trainable activation functions. *Neural Networks*.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115.

Arthur, D. and Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 1027–1035.

Bach, F. R. and Jordan, M. I. (2005). Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, page 33–40, New York, NY, USA. Association for Computing Machinery.

Bandyapadhyay, S., Fomin, F. V., and Simonov, K. (2020). On coresets for fair clustering in metric and euclidean spaces and their applications. *CoRR*, abs/2007.10137.

Bartlett, P., Linder, T., and Lugosi, G. (1998a). The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory*, 44(5):1802–1813.

Bartlett, P., Maiorov, V., and Meir, R. (1998b). Almost linear vc dimension bounds for piecewise polynomial networks. *Advances in neural information processing systems*, 11.

Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.

Bartlett, P. L., Linder, T., and Lugosi, G. (1998c). The minimax distortion redundancy in empirical quantizer design. *IEEE Trans. Inf. Theory*, 44(5):1802–1813.

Bartlett, P. L. and Mendelson, S. (2002). Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482.

Becchetti, L., Bury, M., Cohen-Addad, V., Grandoni, F., and Schwiegelshohn, C. (2019). Oblivious dimension reduction for k-means: beyond subspaces and the johnson-lindenstrauss lemma. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1039–1050.

Berner, J., Grohs, P., Kutyniok, G., and Petersen, P. (2021). The modern mathematics of deep learning. *arXiv preprint arXiv:2105.04026*, pages 86–114.

Bhatt, R. and Dhall, A. (2012). Skin Segmentation. UCI Machine Learning Repository.

Bhatt, U., Ravikumar, P., et al. (2019). Building human-machine trust via interpretability. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 9919–9920.

Bhattacharyya, C., Kannan, R., and Kumar, A. (2022). How many clusters? - an algorithmic answer. In Naor, J. S. and Buchbinder, N., editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 2607–2640. SIAM.

Bianchini, M. and Scarselli, F. (2014). On the complexity of neural network classifiers: A comparison between shallow and deep architectures. *IEEE transactions on neural networks and learning systems*, 25(8):1553–1565.

Biau, G., Devroye, L., and Lugosi, G. (2008a). Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9(9).

Biau, G., Devroye, L., and Lugosi, G. (2008b). On the performance of clustering in hilbert spaces. *IEEE Transactions on Information Theory*, 54(2):781–790.

Blackard, J. (1998). Covertype. UCI Machine Learning Repository.

Bragatto, T., Bucarelli, M. A., Bucarelli, M. S., Carere, F., Geri, A., and Maccioni, M. (2023). False data injection impact on high res power systems with centralized voltage regulation architecture. *Sensors*, 23(5):2557.

Braverman, V., Cohen-Addad, V., Jiang, S. H., Krauthgamer, R., Schwiegelshohn, C., Toftrup, M. B., and Wu, X. (2022). The power of uniform sampling for coresets. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 462–473. IEEE.

Bredon, G. E. (2013). *Topology and geometry*, volume 139. Springer Science & Business Media.

Bucarelli, M. S., Cassano, L., Siciliano, F., Mantrach, A., and Silvestri, F. (2023a). Leveraging inter-rater agreement for classification in the presence of noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3439–3448.

Bucarelli, M. S., Larsen, F. M., Schwiegelshohn, C., and Toftrup, M. B. (2023b). On generalization bounds for projective clustering. In Oh, A., Nauman, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*.

Calandriello, D. and Rosasco, L. (2018). Statistical and computational trade-offs in kernel k-means. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Cannarsa, P. and D'Aprile, T. (2015). *Introduction to Measure Theory and Functional Analysis*. UNITEXT. Springer International Publishing.

Chierichetti, F., Kumar, R., Lattanzi, S., and Vassilvitskii, S. (2017). Fair clustering through fairlets. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5029–5037.

Chitta, R., Jin, R., Havens, T. C., and Jain, A. K. (2011). Approximate kernel k-means: Solution to large scale kernel clustering. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, page 895–903, New York, NY, USA. Association for Computing Machinery.

Chitta, R., Jin, R., and Jain, A. K. (2012). Efficient kernel clustering using random fourier features. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining*, ICDM '12, page 161–170, USA. IEEE Computer Society.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2015a). The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR.

Choromanska, A., LeCun, Y., and Arous, G. B. (2015b). Open problem: The landscape of the loss surfaces of multilayer networks. In *Conference on Learning Theory*, pages 1756–1760. PMLR.

Chou, P. (1994). The distortion of vector quantizers trained on n vectors decreases to the optimum as $O_p(1/n)$. In *Proceedings of 1994 IEEE International Symposium on Information Theory*, pages 457–.

Chowdhary, K. and Chowdhary, K. (2020). Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649.

Clarkson, K. L. and Woodruff, D. P. (2009). Numerical linear algebra in the streaming model. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 205–214.

Clarkson, K. L. and Woodruff, D. P. (2015). Input sparsity and hardness for robust subspace approximation. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 310–329.

Clémençcon, S. (2011). On u-processes and clustering performance. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohen, M. B., Elder, S., Musco, C., Musco, C., and Persu, M. (2015). Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 163–172.

Cohen-Addad, V., Larsen, K. G., Saulpic, D., and Schwiegelshohn, C. (2022a). Towards optimal lower bounds for k-median and k-means coresets. In Leonardi, S. and Gupta, A., editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1038–1051. ACM.

Cohen-Addad, V., Larsen, K. G., Saulpic, D., Schwiegelshohn, C., and Sheikh-Omar, O. A. (2022b). Improved coresets for euclidean k-means. In *NeurIPS*.

Cohen-Addad, V., Saulpic, D., and Schwiegelshohn, C. (2021). Improved coresets and sublinear algorithms for power means in euclidean spaces. *Advances in Neural Information Processing Systems*, 34.

Cohen-Addad, V., Saulpic, D., and Schwiegelshohn, C. (2021a). Improved coresets and sublinear algorithms for power means in euclidean spaces. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 7-10, 2021, Virtual Conference.*

Cohen-Addad, V., Saulpic, D., and Schwiegelshohn, C. (2021b). A new coreset framework for clustering. In Khuller, S. and Williams, V. V., editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021.* ACM.

Cohen-Addad, V. and Schwiegelshohn, C. (2017). On the local structure of stable clustering instances. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 49–60.

Collins, K. M., Bhatt, U., and Weller, A. (2022). Eliciting and learning with soft labels from every annotator.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(2):303–314.

Dalvi, F., Durrani, N., Sajjad, H., Belinkov, Y., Bau, A., and Glass, J. (2019a). What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6309–6317.

Dalvi, F., Nortonsmith, A., Bau, A., Belinkov, Y., Sajjad, H., Durrani, N., and Glass, J. (2019b). Neurox: A toolkit for analyzing individual neurons in neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9851–9852.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27.

Dawid, A. P. and Skene, A. M. (1979). Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):20–28.

Deshpande, A. and Varadarajan, K. R. (2007). Sampling-based dimension reduction for subspace approximation. In Johnson, D. S. and Feige, U., editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 641–650. ACM.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devroye, L., Györfi, L., and Lugosi, G. (2013). *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media.

Dhillon, I. S., Guan, Y., and Kulis, B. (2004). Kernel k-means: Spectral clustering and normalized cuts. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 551–556, New York, NY, USA. Association for Computing Machinery.

Ding, H. (2020). A sub-linear time framework for geometric optimization with outliers in high dimensions. In Grandoni, F., Herman, G., and Sanders, P., editors, *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)*, volume 173 of *LIPIcs*, pages 38:1–38:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Fan, F., Cong, W., and Wang, G. (2018). A new type of neurons for machine learning. *International journal for numerical methods in biomedical engineering*, 34(2):e2920.

Fefferman, C., Mitter, S., and Narayanan, H. (2016). Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049.

Feldman, D. and Langberg, M. (2011). A unified framework for approximating and clustering data. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 569–578.

Feldman, D., Monemizadeh, M., Sohler, C., and Woodruff, D. P. (2010). Coresets and sketches for high dimensional subspace approximation problems. In Charikar, M., editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 630–649. SIAM.

Feldman, D., Schmidt, M., and Sohler, C. (2020). Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM J. Comput.*, 49(3):601–657.

Feng, L., Shu, S., Lin, Z., Lv, F., Li, L., and An, B. (2020). Can cross entropy loss be robust to label noise? In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 2206–2212. International Joint Conferences on Artificial Intelligence Organization. Main track.

Fleiss, J. et al. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Foster, D. J. and Rakhlin, A. (2019). ℓ_∞ vector contraction for rademacher complexity. *CoRR*, abs/1911.06468.

Freeman, C. D. and Bruna, J. (2016). Topology and geometry of half-rectified network optimization. *arXiv preprint arXiv:1611.01540*.

Freeman, C. D. and Bruna, J. (2019). Topology and geometry of half-rectified network optimization. In *5th International Conference on Learning Representations, ICLR 2017*.

Gabrielov, A. and Vorobjov, N. (1995). Complexity of stratification of semi-pfaffian sets. *Discrete & computational geometry*, 14(1):71–91.

Gabrielov, A. and Vorobjov, N. (2004). Complexity of computations with pfaffian and noetherian functions. *Normal forms, bifurcations and finiteness problems in differential equations*, 137:211–250.

Ghorbani, A., Abid, A., and Zou, J. (2019). Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3681–3688.

Ghosh, A., Kumar, H., and Sastry, P. S. (2017). Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 1919–1925. AAAI Press.

Ghosh, A., Manwani, N., and Sastry, P. (2015). Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. (2014). Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*.

Gupta, M. M., Bukovsky, I., Homma, N., Solo, A. M., and Hou, Z.-G. (2013). Fundamentals of higher order neural networks for modeling and simulation. In *Artificial Higher Order Neural Networks for Modeling and Simulation*, pages 103–133. IGI Global.

Han, B., Yao, Q., Liu, T., Niu, G., Tsang, I. W., Kwok, J. T., and Sugiyama, M. (2020). A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hatcher, A. (2002). Algebraic topology, cambridge univ. *Press, Cambridge*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. (2018). Using trusted data to train deep networks on labels corrupted by severe noise. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Heo, B., Lee, M., Yun, S., and Choi, J. Y. (2019). Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787.

Horn, R. A. and Johnson, C. R. (2012). *Matrix Analysis*. Cambridge University Press, USA, 2nd edition.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Huang, L., Li, J., and Wu, X. (2022). Towards optimal coreset construction for (k, z)-clustering: Breaking the quadratic dependency on k. *CoRR*, abs/2211.11923.

Huang, L., Sudhir, K., and Vishnoi, N. K. (2021). Coresets for time series clustering.

Karpinski, M. and Macintyre, A. (1997). Polynomial bounds for vc dimension of sigmoidal and general pfaffian neural networks. *Journal of Computer and System Sciences*, 54(1):169–176.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. (2022). Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41.

Khetan, A., Lipton, Z. C., and Anandkumar, A. (2017). Learning from noisy singly-labeled data.

Khovanskii, A. G. (1991). *Fewnomials – Translations of Mathematical Monographs 88*. American Mathematical Society.

Khurana, D., Koli, A., Khatter, K., and Singh, S. (2023). Natural language processing: State of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744.

Klein, P. N. and Young, N. E. (2015). On the number of iterations for dantzig-wolfe optimization and packing-covering approximation algorithms. *SIAM J. Comput.*, 44(4):1154–1172.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25*, pages 1097–1105.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kulkarni, R. V. and Venayagamoorthy, G. K. (2009). Generalized neuron: Feedforward and recurrent architectures. *Neural networks*, 22(7):1011–1017.

Kumar, V. S. A., Arya, S., and Ramesh, H. (2000). Hardness of set cover with intersection 1. In Montanari, U., Rolim, J. D. P., and Welzl, E., editors, *Automata, Languages and Programming, 27th International Colloquium, ICALP 2000, Geneva, Switzerland, July 9-15, 2000, Proceedings*, volume 1853 of *Lecture Notes in Computer Science*, pages 624–635. Springer.

Lauer, F. (2020). Risk bounds for learning multiple components with permutation-invariant losses. In Chiappa, S. and Calandra, R., editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 1178–1187. PMLR.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Levrard, C. (2015). Nonasymptotic bounds for vector quantization in hilbert spaces. *The Annals of Statistics*, 43(2).

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. (2018). Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31.

Linder, T. (2000). On the training distortion of vector quantizers. *IEEE Transactions on Information Theory*, 46(4):1617–1623.

Linder, T., Lugosi, G., and Zeger, K. (1994). Rates of convergence in the source coding theorem, in empirical quantizer design, and in universal lossy source coding. *IEEE Transactions on Information Theory*, 40(6):1728–1740.

Liu, F., Huang, X., Chen, Y., and Suykens, J. A. K. (2020). Random features for kernel approximation: A survey on algorithms, theory, and beyond.

Liu, Y. (2021). Refined learning bounds for kernel and approximate \$k\$-means. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 6142–6154.

Liu, Y., Liao, S., Jiang, S., Ding, L., Lin, H., and Wang, W. (2019). Fast cross-validation for kernel-based algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136.

Maennel, H., Bousquet, O., and Gelly, S. (2018). Gradient descent quantizes relu network features. *arXiv preprint arXiv:1803.08367*.

Makarychev, K., Makarychev, Y., and Razenshteyn, I. P. (2019). Performance of johnson-lindenstrauss transform for k-means and k-medians clustering. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1027–1038.

Meir, R. and Zhang, T. (2003). Generalization error bounds for bayesian mixture algorithms. *J. Mach. Learn. Res.*, 4:839–860.

Menon, A., Rooyen, B. V., Ong, C. S., and Williamson, B. (2015). Learning from corrupted binary labels via class-probability estimation. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 125–134, Lille, France. PMLR.

Menon, A. K., Rawat, A. S., Reddi, S. J., and Kumar, S. (2020). Can gradient clipping mitigate label noise? In *International Conference on Learning Representations*.

Mettu, R. R. and Plaxton, C. G. (2004). Optimal time bounds for approximate clustering. *Mach. Learn.*, 56(1-3):35–60.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012). *Foundations of Machine Learning*. The MIT Press.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Muselli, M. and Ruffino, F. (2004). In *Consistency of Empirical Risk Minimization for Unbounded Loss Functions*, pages 261–270.

Naitzat, G., Zhitnikov, A., and Lim, L.-H. (2022). Topology of deep neural networks. *J. Mach. Learn. Res.*, 21(1).

Nam, W.-J., Gur, S., Choi, J., Wolf, L., and Lee, S.-W. (2020). Relative attributing propagation: Interpreting the comparative contributions of individual units in deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2501–2508.

Narayanan, S. and Nelson, J. (2019). Optimal terminal dimensionality reduction in euclidean space. In Charikar, M. and Cohen, E., editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1064–1069. ACM.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., and Tewari, A. (2013). Learning with noisy labels. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Nguyen, Q. and Hein, M. (2017). The loss surface of deep and wide neural networks. In *International conference on machine learning*, pages 2603–2612. PMLR.

Oglic, D. and Gärtner, T. (2017). Nyström method with kernel k-means++ samples as landmarks. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2652–2660. PMLR.

Oneto, L., Navarin, N., Biggio, B., Errica, F., Micheli, A., Scarselli, F., Bianchini, M., Demetrio, L., Bongini, P., Tacchella, A., and Sperduti, A. (2022). Towards learning trustworthily, automatically, and with guarantees on graphs: An overview. *Neurocomputing*, 493:217–243.

OpenAI (2023). Gpt-4 technical report.

Ostrovsky, R., Rabani, Y., Schulman, L. J., and Swamy, C. (2012). The effectiveness of lloyd-type methods for the k-means problem. *J. ACM*, 59(6):28:1–28:22.

Patrini, G., Rozza, A., Krishna, A., Nock, R., and Qu, L. (2017). Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952.

Peterson, J., Battleday, R., Griffiths, T., and Russakovsky, O. (2019). Human uncertainty makes classification more robust. In *Proceedings - 2019 International Conference on Computer Vision, ICCV 2019*, Proceedings of the IEEE International Conference on Computer Vision, pages 9616–9625, United States. Institute of Electrical and Electronics Engineers Inc.

Pollard, D. (1981). Strong Consistency of K-Means Clustering. *The Annals of Statistics*, 9(1):135 – 140.

Pollard, D. (1982a). A Central Limit Theorem for k-Means Clustering. *The Annals of Probability*, 10(4):919 – 926.

Pollard, D. (1982b). Quantization and the method of k-means. *IEEE Trans. Inf. Theory*, 28(2):199–204.

Potter, J. E. (1966). Matrix quadratic solutions. *SIAM Journal on Applied Mathematics*, 14(3):496–501.

Prabhavalkar, R., Hori, T., Sainath, T. N., Schlüter, R., and Watanabe, S. (2023). End-to-end speech recognition: A survey. *arXiv preprint arXiv:2303.03329*.

Purpura, A., Silvello, G., and Susto, G. A. (2022). Learning to rank from relevance judgments distributions.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Searching for activation functions. *arXiv preprint arXiv:1710.05941*.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3.

Raykar, V. C., Yu, S., Zhao, L. H., Valadez, G. H., Florin, C., Bogoni, L., and Moy, L. (2010). Learning from crowds. *Journal of Machine Learning Research*, 11(43):1297–1322.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386.

Rudi, A. and Rosasco, L. (2016). Generalization properties of learning with random features.

Rudin, W. (1976). *Principles of mathematical analysis*. International series in pure and applied mathematics. McGraw-Hill, New York.

Rudin, W. (1987). *Real and Complex Analysis*. McGraw-Hill, New York.

Rudra, A. and Wootters, M. (2014). Every list-decodable code for high noise has abundant near-optimal rate puncturings. In Shmoys, D. B., editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 764–773. ACM.

Safran, I. and Shamir, O. (2018). Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pages 4433–4441. PMLR.

Sarlós, T. (2006). Improved approximation algorithms for large matrices via random projections. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 143–152.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2008). The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

Scarselli, F., Tsoi, A. C., and Hagenbuchner, M. (2018). The vapnik–chervonenkis dimension of graph and recursive neural networks. *Neural Networks*, 108:248–259.

Schlimmer, J. (1987). Mushroom. UCI Machine Learning Repository.

Schmidt, M., Schwiegelshohn, C., and Sohler, C. (2019). Fair coresets and streaming algorithms for fair k-means. In *Approximation and Online Algorithms - 17th International Workshop, WAOA 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers*, pages 232–251.

Shawe-Taylor, J., Williams, C. K. I., Cristianini, N., and Kandola, J. S. (2005). On the eigenspectrum of the gram matrix and the generalization error of kernel-pca. *IEEE Trans. Inf. Theory*, 51(7):2510–2522.

Sherman, J. and Morrison, W. (1949). Abstracts of Papers. *The Annals of Mathematical Statistics*, 20(4):620 – 624.

Shyamalkumar, N. D. and Varadarajan, K. R. (2007). Efficient subspace approximation algorithms. In Bansal, N., Pruhs, K., and Stein, C., editors, *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 532–540. SIAM.

Siciliano, F., Bucarelli, M. S., Tolomei, G., and Silvestri, F. (2022). Newron: A new generalization of the artificial neuron to enhance the interpretability of neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 01–17.

Siciliano, F., Magister, L. C., Bucarelli, M. S., Barbiero, P., Silvestri, F., and Lio, P. (2023). Explaining neural networks using a ruleset based on interpretable concepts. In *Submitted to EPJ Data Science.*

Sohler, C. and Woodruff, D. P. (2018). Strong coresets for k-median and subspace approximation: Goodbye dimension. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 802–813.

Song, H., Kim, M., Park, D., Shin, Y., and Lee, J.-G. (2022). Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–19.

Stein, E. M. and Shakarchi, R. (2005). *Real analysis: measure theory, integration, and Hilbert spaces.* Princeton lectures in analysis. Princeton Univ. Press, Princeton, NJ.

Sun, S., Cao, Z., Zhu, H., and Zhao, J. (2019). A survey of optimization methods from a machine learning perspective. *IEEE transactions on cybernetics*, 50(8):3668–3681.

Telyatnikov, L., Bucarelli, M. S., Bernardez, G., Zaghen, O., Scardapane, S., and Lio, P. (2023). Hypergraph neural networks through the lens of message passing: A common perspective to homophily and architecture design. *arXiv preprint arXiv:2310.07684*.

Tewari, A. and Bartlett, P. L. (2007). On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8(5).

Trappolini, G., Cassarà, G., Bucarelli, M. S., and Silvestri, F. (2023). A simple yet tough to beat baseline to counterfactually explain gnns. In *Submitted to AISTATS 2024*.

Uma, A., Fornaciari, T., Hovy, D., Paun, S., Plank, B., and Poesio, M. (2020). A case for soft loss functions. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, 8(1):173–177.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

Venturi, L., Bandeira, A. S., and Bruna, J. (2018). Spurious valleys in two-layer neural network optimization landscapes. *arXiv preprint arXiv:1802.06384*.

Venturi, L., Bandeira, A. S., and Bruna, J. (2019). Spurious valleys in one-hidden-layer neural network optimization landscapes. *Journal of Machine Learning Research*, 20:133.

Vershynin, R. (2012). Introduction to the non-asymptotic analysis of random matrices. In Eldar, Y. C. and Kutyniok, G., editors, *Compressed Sensing*, pages 210–268. Cambridge University Press.

Wang, S., Gittens, A., and Mahoney, M. W. (2019a). Scalable kernel k-means clustering with nystrom approximation: Relative-error bounds. *Journal of Machine Learning Research*, 20(12):1–49.

Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., and Bailey, J. (2019b). Symmetric cross entropy for robust learning with noisy labels. In *IEEE International Conference on Computer Vision*.

Wani, F. A., Bucarelli, M. S., and Silvestri, F. (2023). Combining distance to class centroids and outlier discounting for improved learning with noisy labels. *arXiv preprint arXiv:2303.09470*.

Wei, J., Zhu, Z., Luo, T., Amid, E., Kumar, A., and Liu, Y. (2022). To aggregate or not? learning with separate noisy labels.

Wheeden, Richard, L. and Zygmund, A. (2015). *Measure and integral: An introduction to real analysis*. CRC Press. Second ediction.

Williams, F., Trager, M., Panozzo, D., Silva, C., Zorin, D., and Bruna, J. (2019). Gradient dynamics of shallow univariate relu networks. *Advances in Neural Information Processing Systems*, 32:8378–8387.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., and Sugiyama, M. (2019). Are anchor points really indispensable in label-noise learning? In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yang, Y., Morillo, I. G., and Hospedales, T. M. (2018). Deep neural decision trees. *arXiv preprint arXiv:1806.06988.*

Yao, Y., Liu, T., Han, B., Gong, M., Deng, J., Niu, G., and Sugiyama, M. (2020). Dual t: Reducing estimation error for transition matrix in label-noise learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Yin, R., Liu, Y., Lu, L., Wang, W., and Meng, D. (2020). Divide-and-conquer learning with nyström: Optimal rate and algorithm. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6696–6703.

Zell, T. (1999). Betti numbers of semi-pfaffian sets. *Journal of Pure and Applied Algebra*, 139(1):323–338.

Zell, T. P. (2003). *Quantitative study of semi-Pfaffian sets.* PhD thesis, Purdue University.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*. PMLR.

Zhang, M., Lee, J., and Agarwal, S. (2021). Learning from noisy labels with no change to the training process. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12468–12478. PMLR.

Zhang, X. and Liao, S. (2019). Incremental randomized sketching for online kernel learning. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7394–7403. PMLR.

Zhang, Z. and Sabuncu, M. R. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8792–8802, Red Hook, NY, USA. Curran Associates Inc.

Zhu, Z., Song, Y., and Liu, Y. (2021). Clusterability as an alternative to anchor points when learning with noisy labels.

Zhu, Z., Wang, J., and Liu, Y. (2022). Beyond images: Label noise transition matrix estimation for tasks with lower-quality features. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 27633–27653. PMLR.