EPFL

Thèse n° 10 519

Fundamental Limits in Statistical Learning Problems: Block Models and Neural Networks

Présentée le 7 juillet 2023

Faculté des sciences de base
Chaire de Mathématiques de la science des données (SB/IC)
Programme doctoral en mathématiques

pour l'obtention du grade de Docteur ès Sciences

par

Elisabetta CORNACCHIA

École
polytechnique
fédérale
de Lausanne

2023

“Il fatto che l’attività svolta in modo così imperfetto sia stata e sia tuttora per me fonte inesauribile di gioia mi fa ritenere che l’imperfezione nell’eseguire il compito che ci siamo prefissi, o ci è stato assegnato, sia più consona alla natura umana così imperfetta che non la perfezione.”

"The fact that the activity carried out in such an imperfect way has been and still is an inexhaustible source of joy for me leads me to believe that imperfection in performing the task we have set ourselves, or have been assigned, is more suitable for human nature, which is so imperfect, than perfection."

— Rita Levi-Montalcini, *Elogio dell’Imperfezione*.

Ai miei genitori.

To my parents.

Acknowledgements

First of all, I would like to express my gratitude to my advisor, Prof. Emmanuel Abbé, who guided me with support, patience and remarkable expertise from my very first steps in research. I am extremely grateful for the time and effort that he dedicated to me, for the scientific and personal discussions, and for renewing my motivation when I felt discouraged. I had the opportunity to work on diverse and exciting topics, participate in collaborations, attend schools, and undertake research visits. These experiences allowed me to immerse myself in the world of research, and I am indebted for these opportunities.

I am also thankful to Professors Lénaïc Chizat, Laurent Massoulié, and Nati Srebro for accepting to be part of the jury at my oral exam and for their comments and feedback. I am also grateful to Professor Friedrich Eisenbrand for presiding over the jury.

Furthermore, I am grateful to Professor Sasha Rakhlin for accepting me as a visiting student in his group at MIT, and Professors Elchanan Mossel, Yury Polyanskiy, and Bin Yu for the research discussions we had during my visit to the US. This experience shaped my motivation.

I would like to thank the current and previous members of the MDS lab who participated in this adventure: Aryo, for the projects we have worked on together and for being a happy and reassuring presence in the office during this last semester; Colin, for the discussions on block models; Enric, for sharing his incredible energy and enthusiasm during his time in Lausanne and my time in Boston; Jan, for his constant support and for teaching me a lot of math; Ido, for the research discussions during my first year; Raphaël, for his support and encouraging words, and for waiting patiently for me to finish my lunches. A special thanks goes to Ariane, who handled all the administrative work with impressive efficiency and professionalism. My trips gave her quite a lot of work, and I am very grateful for her time and help.

I am also grateful to the other researchers with whom I had the chance to collaborate during these years: Samy Bengio, Cédric Gerbelot, Yuzhou Gu, Jon Kleinberg, Bruno Loureiro, Christopher Marquis, Francesca Mignacco, Neta Singer, Rodrigo Veiga, Lenka Zdeborová, and Chiyuan Zhang.

I would like to thank all the people on the second floor of the math building, in particular: Alice, for organizing amazing hikes in Switzerland and California; Andrea, for his support, friendship, and optimism from day one; Eva, for the long chats; Giacomo, for the coffee breaks and for answering my many questions about exam organization; Ondine, for her constant happiness and for all the chats and laughs. I am also thankful to the people of the FLAIR community for the discussions and pizza seminars.

I consider myself incredibly fortunate to have found an amazing group of friends who have made the past years an unforgettable journey. I am thankful to Francesca B., Gloria, Lucia, and Siria for the many moments and trips we have shared together and for always making me feel at home. I am grateful to Lorenzo for cooking the best pizza in town, to Stefano for regularly calling from anywhere in the world, to Leo P. for the study sessions during the lockdown, and to Edo and Leo S. for the ski days. I am thankful to Francesca M. for sharing her passion, knowledge, and energy with me since high school, and to Alicia, Arnau, Chicco, Davide, Giovanni, Guido, Jésus, Laura, Luca, Marco, Matteo, Max, Nicola, Pierre, Sara, and Till for the many evenings, barbecues, beers, and laughs. I am thankful to the flatmates I had during these years: Giulia M., Valentina, and Aurora, for creating a peaceful and warm atmosphere at home. I am also grateful to Beba, Giulia C., and Tea for their warm friendship despite the distance.

Lastly, a very special thanks goes to my family. I express my deepest gratitude to my parents, Giorgio and Valeria, for their constant love and encouragement. I cannot express the importance of their support in realizing this thesis, and this work is dedicated to them. I am grateful to my brothers, Lorenzo and Alessandro, and my cousins, Federica, Jacopo, and Greta, for warmly welcoming me with dinners and games every time I returned to Torino. I feel extremely lucky to have such a loving home. I would also like to thank my grandparents, Nonno Angelo, for always believing in me, and Nonna Anna, for her parmigianas and for keeping me company during the long drives over the phone.

Lausanne, 14th June 2023 Elisabetta Cornacchia

Abstract

This thesis focuses on two selected learning problems: 1) statistical inference on graphs models, and, 2) gradient descent on neural networks, with the common objective of defining and analysing the measures that characterize the fundamental limits.

In the first part of the thesis, we consider spin synchronization problems on graphs, which consist of reconstructing a vector of n independent spins living on the vertices of a graph, based on noisy observations of their interactions on the edges of the graph. In particular, we consider synchronization models with erasure (BEC) side-information, where the spins of a small fraction of nodes are revealed, and investigate how the addition of such side-information influences the correlations between spins at distant sites. We show that on trees, whenever spins at distant sites are nearly independent given the edge observations, then they are still nearly independent given the edge observations and the side-information. We conjecture this to hold for any graph. On the other hand, (Kanade et al., 2014) conjectured that, on regular trees and on Galton-Watson trees, whenever any small fraction of node labels are revealed, the boundary at infinite depth becomes ineffective for detecting the root bit, even in the reconstruction regime. We explain how this can be used for computing the limiting entropy of the sparse Stochastic Block Model (SBM) with two symmetric communities. Finally, we show that the latter conjecture does not hold for every tree.

In the second part of the thesis, we consider the problem of learning Boolean target functions with gradient descent (GD) on fully connected neural networks. We introduce the notion of "Initial Alignment" (INAL) between a neural network at initialization and a target function and prove that if a network and target do not have a noticeable INAL, then noisy gradient descent on a fully connected network with i.i.d. Gaussian initialization cannot learn the target in polynomial time. We show that for finite depth networks trained with the correlation loss, the result can be extended beyond Boolean inputs. Moreover, we prove that in a similar setting, the generalization error can be lower-bounded in terms of the noise-stability of the target function, supporting a conjecture made in (Zhang et al., 2021).

We then show that in the distribution shift setting, when the data withholding corresponds to freezing a single feature, the generalisation error admits a tight characterisation in terms of the Boolean influence for several relevant architectures. This is shown on linear models and supported experimentally on other models such as MLPs and Transformers. In particular,

this puts forward the hypothesis that for such architectures and for learning logical functions, GD tends to have an implicit bias towards low-degree representations.
Finally, we consider a ‘curriculum learning’ (CL) strategy for learning k-parities on d bits of a binary string. We show that a wise choice of training examples, involving two or more product distributions, allows to learn k-parities in $d^{O(1)}$ time with a fully connected neural network trained with GD. We further show that for another class of functions - namely the ‘Hamming mixtures’ - CL strategies involving a bounded number of product distributions are not beneficial.

Key words: statistical learning, fundamental limits, complexity measures, stochastic block models, synchronization on graphs, neural networks.

Résumé

Cette thèse se concentre sur deux problèmes d'apprentissage : 1) l'inférence statistique de modèles sur graphes et 2) la descente de gradient sur les réseaux de neurones, avec l'objectif commun de définir et d'analyser les mesures qui caractérisent les limites fondamentales.

Dans la première partie de la thèse, nous examinons les problèmes de synchronisation de spins sur les graphes, qui consistent à reconstruire un vecteur de n spins indépendants vivant sur les sommets d'un graphe, en fonction des observations bruyantes de leurs interactions sur les arêtes du graphe. En particulier, nous considérons des modèles de synchronisation avec une information latérale d'effacement (BEC), où les spins d'une petite fraction de nœuds sont révélés, et étudions comment l'ajout de cette information latérale influence les corrélations entre les spins sur des sites éloignés. Nous montrons que sur les arbres, chaque fois que les spins sur des sites éloignés sont presque indépendants étant donné les observations des arêtes, ils restent presque indépendants étant donné les observations des arêtes et de l'information latérale. Nous émettons l'hypothèse que cela est vrai pour chaque graphe. D'autre part, nous montrons que la conjecture de (KANADE et al., 2014) - selon laquelle sur les arbres réguliers et les arbres Galton-Watson, lorsqu'une petite fraction d'étiquettes de nœuds est révélée, la frontière à profondeur infinie devient inefficace pour détecter le bit à la racine de l'arbre, même dans le régime de reconstruction - ne tient pas pour chaque arbre.

Dans la deuxième partie de la thèse, nous considérons le problème de l'apprentissage de fonctions objectifs booléennes avec la descente de gradient sur des réseaux de neurones entièrement connectés. Nous introduisons la notion d'"alignement initial" (INAL) entre un réseau neuronal à l'initialisation et une fonction objectif et nous prouvons que si un réseau et une objectif n'ont pas d'INAL perceptible, alors la descente de gradient sur un réseau entièrement connecté avec initialisation gaussienne i.i.d. ne peut pas apprendre l'objectif en temps polynomial. Nous montrons que pour les réseaux de profondeur finie entraînés avec la 'correlation loss', le résultat peut être étendu au-delà des entrées booléennes. De plus, nous prouvons que dans un cadre semblable, l'erreur de généralisation peut être bornée inférieurement en termes de stabilité au bruit de la fonction objectif, soutenant ainsi une conjecture qui apparaît dans (ZHANG et al., 2021).

Nous montrons ensuite que dans le cadre de changements de distribution, lorsque la rétention de données correspond à la fixation d'une seule caractéristique, l'erreur de généralisation peut être caractérisée en termes d'influence booléenne pour plusieurs architectures. Cela est démontré sur des modèles linéaires et est soutenu expérimentalement sur d'autres modèles tels que le Multi Layer Perceptron et Transformers. En particulier, cela avance l'hypothèse que pour de telles architectures et pour l'apprentissage de fonctions logiques, la descente de gradient tend à avoir un biais implicite vers des représentations de faible degré. Enfin, nous considérons une stratégie d'apprentissage par curriculum (CL) pour apprendre la classe des k-parités sur d bits d'une chaîne binaire avec un réseau de neurones entièrement connecté entraîné avec la descente de gradient stochastique. Nous montrons qu'un choix judicieux d'exemples d'entraînement, impliquant deux ou plusieurs distributions de produits, permet l'apprentissage de cette classe de fonctions en temps $d^{O(1)}$ donnée une distribution uniforme. Nous montrons également que pour une autre classe de fonctions - les 'Hamming mixtures' - les stratégies de CL impliquant un nombre borné de distributions de produits ne sont pas bénéfiques.

Mots cléfs : apprentissage statistique, limites fondamentales, mesures de complexité, modèles de blocs stochastiques, synchronisation sur les graphes, réseaux de neurones.

Sommario

Questa tesi si concentra su due problemi di apprendimento: 1) inferenza statistica sui modelli su grafi e 2) discesa del gradiente sulle reti neurali, con l'obiettivo comune di definire ed analizzare le misure che caratterizzano i limiti fondamentali.

Nella prima parte della tesi, consideriamo problemi di sincronizzazione di spin su grafi, che consistono nella ricostruzione di un vettore di n spin indipendenti situati sui vertici di un grafo, basandosi su osservazioni con rumore delle loro interazioni sugli archi del grafo. In particolare, consideriamo modelli di sincronizzazione con informazione laterale di tipo BEC, in cui gli spin di una piccola frazione di nodi vengono rivelati, e indaghiamo come l'aggiunta di tale informazione laterale influisca sulle correlazioni tra gli spin in siti lontani. Dimostriamo che sui grafi ad albero, quando gli spin in siti lontani sono approssimativamente indipendenti rivelate le osservazioni sugli archi, allora sono ancora approssimativamente indipendenti date le osservazioni sugli archi e l'informazione laterale. Ipotizziamo che ciò sia vero per qualsiasi grafo. D'altra parte, (Kanade et al., 2014) ipotizzano che, sui grafi ad albero regolari e ad albero Galton-Watson, quando viene rivelata una frazione arbitrariamente piccola di nodi, i nodi a profondità infinita diventano irrilevanti per rilevare lo spin alla radice, anche nel regime di ricostruzione. Spieghiamo come questo possa essere utilizzato per calcolare l'entropia nello stochastic block model (SBM) nel regime sparso con due comunità simmetriche. Infine, dimostriamo che quest'ultima congettura non vale per ogni grafo ad albero.

Nella seconda parte della tesi, consideriamo il problema di apprendere funzioni obiettivo Booleane con il metodo della discesa del gradiente (GD) su reti neurali completamente connesse. Introduciamo la nozione di "Allineamento Iniziale" (INAL) tra una rete neurale all'inizializzazione e una funzione obiettivo e dimostriamo che se una rete e la funzione obiettivo non presentano un INAL evidente, allora la discesa del gradiente su una rete completamente connessa con inizializzazione i.i.d Gaussiana non può imparare l'obiettivo in tempo polinomiale. Mostriamo che per reti a profondità finita addestrate con funzione costo data dalla correlazione, il risultato può essere esteso oltre gli input Booleani. Inoltre, dimostriamo che in un contesto simile, l'errore di generalizzazione può essere inferiormente limitato dalla stabilità al rumore della funzione obiettivo, supportando una congettura formulata in (Zhang et al., 2021).

Successivamente mostriamo che in un contesto di shift di distribuzione, quando una singola coordinata é fissata durante l'addestramento, l'errore di generalizzazione ammette una caratterizzazione precisa in termini dell'influenza Booleana per diverse architetture rilevanti. Questo é dimostrato per modelli lineari, e supportato sperimentalmente su altre architetture, come Multi Layer Perceptron e Transformers. In particolare, ne consegue l'ipotesi che per tali architetture e per funzioni obiettivo Booleane, il metodo della discesa del gradiente ha un bias implicito verso rappresentazioni di grado basso.
Infine, consideriamo una strategia di addestramento con curriculum (CL) per imparare parità di grado k su d bits di una stringa binaria. Dimostriamo che una scelta saggia di esempi durante il training, che può comportare due o più distribuzioni prodotto, consente di imparare le funzioni parità di grado k in tempo $d^{O(1)}$ con una rete completamente connessa allenata con la discesa del gradiente stocastica. Mostriamo, inoltre, che per un'altra classe di funzioni, le 'Hamming mixtures', le strategie con curriculum con un numero limitato di distribuzioni prodotto non portano benefici.

Parole chiave: apprendimento statistico, limiti fondamentali, misure di complessità, modelli a blocchi stocastici, sincronizzazione su grafi, reti neurali.

Contents

List of Figures

1 Introduction

Data science is a rapidly evolving field concerned with extracting valuable information from large and complex datasets. This challenge affects several areas of today's data-driven world. For instance, product recommendation systems detect user preferences by analyzing past purchases, browsing behaviors, and demographic information to provide personalized recommendations; similarly, financial pricing entails analyzing historical trends, supply and demand data, and macroeconomic factors to predict commodity price changes.

Statistical inference is a fundamental concept in statistics that involves drawing conclusions about the characteristics of an underlying probability distribution based on observations of data. This process involves applying mathematical and statistical techniques to the observed data to estimate the unknown parameters of the underlying distribution. Formally, one considers a set of unknown variables $\{\xi_i\}_{i=1,...,N}$ and a set of observations (or data) $\{s_j\}_{j=1,..,M}$. Data can consist of noisy observations of the variables themselves, or interactions of subsets of variables, or they may contain indirect information about the variables. As examples, we introduce two inference settings that will be relevant in this thesis.

Graphs Models. Graphs models are a powerful tool for modeling complex systems, such as social networks or biological systems. The underlying structure is given by a graph G, with n vertices and m edges. The vertices represent the different components of the system, while the edges capture the relationships between them. In inference problems on graphs, it is assumed that vertices are assigned unknown variables $\{\xi_i\}_{i=1,...,n}$, that the learner has to retrieve based on observations $\{s_j\}_{j=1,..,m}$ on the edges. For example, in social networks, the ξ_i variables can model the community cluster that each agent belongs to, while the s_j observations serve as indicators of whether two agents are friends or share a common opinion. Overall, graphs models provide a framework for modeling complex systems and performing inference on them. By leveraging the relationships between different components of the system, one can gain insights into their behavior and predict future outcomes with greater accuracy.

Supervised Learning. Supervised learning is a central approach in machine learning. It is widely used in various applications, such as image classification, speech recognition, and natural language processing. The goal is to learn a mapping between input and output by observing pairs of examples (x, y), where x is the input (or *feature*) vector and y is the corresponding output (or *label*). The underlying assumption is that the relationship between input and output can be modeled by a function belonging to a parametric family of models $\{f_\theta\}_{\theta\in\Theta}$. The key objective, thus, reduces to find the optimal parameters $\theta^* \in \Theta$ that capture the relationship between the pairs of examples observed. In the statistical inference framework, the optimal parameters θ^* represent the unknown variables that the learner has to retrieve, based on the pairs of examples (x, y), which represent the observations. The optimal values θ^* are typically obtained by minimizing a *loss function* that quantifies the difference between the model's predictions and the actual output values observed. In the machine learning community, the process of finding the θ^* is commonly referred to *training* or *learning*. The *training set* is the set of observations used to train the model, and the performance of the learned model is typically evaluated on a separate set of labeled examples, called the *test set*. The performance of the model on the test set provides a measure of how well the model generalizes to new, unseen data.

There are two main aspects about inference problems that are relevant for us:

1. *Information-theoretic aspect.* Do the available observations provide sufficient information to recover the underlying variables?
2. *Computational aspect.* Can the recovery be carried out in an algorithmically *efficient* way?

Although these two aspects are connected in certain ways (for instance, if recovery is information-theoretically impossible, it is also impossible to do efficiently), they are fundamentally different in nature. From an information-theoretic point of view, the first question seeks to understand whether a given task can be solved regardless of complexity or algorithmic considerations. It is a question that lies at the intersection of statistics and information theory. On the other hand, the second question concerns the feasibility of implementing the algorithms used for inference in a reasonable amount of time, and, therefore, it falls in the scopes of computer science and complexity theory. In the past decades, these two questions were typically considered separately, as the number of variables to be estimated was usually small and inference problems were not typically concerned with computational tractability. However, with the recent increase in computational power and memory, it is now possible to perform inference on problems with billions of parameters. As a result, it has become increasingly important to consider both statistical and computational efficiency together. This requires the development of algorithms that can scale to large data sets and complex models while also being efficient in terms of computation time and memory usage.

This thesis aims to explore the information-theoretic and computational limits of modern inference problems. We consider two selected types of problems:

1. *Block Models.* These models belong to the set of graphs models introduced above and they can model inference problems on social or biological populations, where one tries to detect information about individuals based on their interactions;
2. *Neural Networks.* This family of models have gained immense popularity in recent years due to their ability to model complex, nonlinear relationships between inputs and outputs. We explore the effectiveness of neural networks at performing some specific supervised learning tasks.

We will present and discuss these two classes separately in the next sections. Our goal is to analyse the statistical and computational complexity of these problems from a mathematical point of view. For this reason, we restrict our attention to stylized settings that make the analysis tractable.

1.1 Block Models

Many datasets in modern applications can be represented as a graph of interacting agents. For instance, this is the case in sociology (Goldenberg et al., 2010; Mitchell, 1974; Newman and Park, 2003; Scott, 2002), epidemiology (Eames and Read, 2008; Morris, 1993), product recommendation systems (Chen et al., 2013; Sahebi and Cohen, 2011; Zanin et al., 2008), protein folding (Dokholyan et al., 2002; Rao and Caflisch, 2004), webpage sorting (Kim et al., 2010; Kumar et al., 1999), and many others. A natural task of interest with such graphs is understanding which agents are 'similar'. *Community detection* refers to the problem of finding similarities among vertices in a graph, based on measurements of local interactions, and, as such, it is a central problem in the data science community. A popular stylized model for studying community detection on random graphs is the *Stochastic Block Model (SBM).*

1.1.1 The Stochastic Block Model (SBM)

The history of the SBM goes back to the 80's, when it appeared independently in different scientific communities, with different names. The SBM terminology appeared in the machine learning and statistics community (Holland et al., 1983), while in the theoretical computer science literature the model appeared under the name of 'planted partition model' (Boppana, 1987; Bui et al., 1984) and in the mathematics literature it is typically called the 'inhomogeneous random graphs' model (Bollobás et al., 2007).

In the SBM, a vector of community assignments X and a random graph G are drawn according to a joint distribution. Firstly, each vertex is assigned to a group or "community" based on a pre-determined probability distribution. Then, between each pair of vertices, an edge is drawn with probability depending on the communities of its two end points, independently of other edges. Community detection in the SBM consists of retrieving the ground truth community assignment X, based on observation of the graph structure G

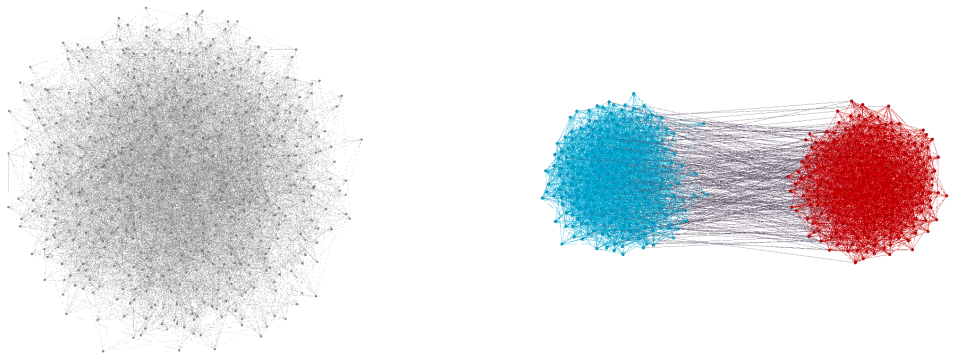

Figure 1.1: Community detection in an SBM with 2 symmetric communities. The two pictures represent the same graph, with random layout (left) and ForceAtlas layout[I] (right). We seek for an estimator that upon observation of the graph structure (left) retrieves the planted clustering of the vertices (right).

(Figure 1.1). Because of its planted community structure and its simple definition, the SBM represents a good benchmark to study community detection in a formal frame. On the other hand, its fit to realistic data is not necessarily adequate, and several variations of the standard SBM (e.g. edge labelled, overlapping communities) has been proposed to fill this gap.

Few natural questions arise: *Which algorithms can recover the community assignments? Do they recover the label of all vertices? What is the optimal fraction of vertices that can be recovered?* The answers depend on the parameters of the model. In the literature, *exact recovery* refers to the problem of recovering the full vector of communities. *Partial recovery* refers to the problem of recovering the assignments of a fraction of the vertices. *Weak recovery* refers to the problem of detecting the community of a non-trival fraction of the vertices (i.e. with accuracy better than guessing).

1.1.2 The SBM Entropy

In this thesis, we make progress on the study of the *fundamental limits* of weak recovery in the simplest case of SBM with two symmetric communities in the sparse regime. In this simplified SBM, the vector of communities X is drawn uniformly at random in $\{\pm 1\}^n$ and an n-vertex graph is drawn by connecting vertices having the same values in X with probability a/n and vertices having different values in X with probability b/n, for some $a, b \in \mathbb{R}_+$. We focus on the information-theoretic limits: we are interested in understanding the optimal error achievable by *any* algorithm, irrespective of its computational cost.

The question of whether it is possible (or not) to recover the communities with accuracy

[I]The ForceAtlas layout algorithm used in Gephi works by simulating physical forces, where nodes repel each other and edges attract nodes together.

better than guessing, has been fully closed in the two-community symmetric SBM (Massoulié, 2014; Mossel et al., 2018). In particular, it has been shown that weak recovery is solvable if and only if

$$\frac{(a-b)^2}{2(a+b)} > 1, \tag{1.1}$$

establishing, thus, a sharp threshold phenomena. The quantity on the left hand side of (1.1) is the signal-to-noise ratio (SNR), which quantify the "signal" of the community structure to the "noise" of the random connections in the graph.

Rather than understanding whether weak recovery is possible (or not), we focus here on a more detailed question. We want to characterize the *optimal* fraction of labels that can be recovered by any algorithm, for given a, b in the limit of $n \to \infty$. In other words, we want to quantify the amount of information that can be recovered about the communities at any value of the SNR. Several measures of community signal in the graph can be used: the optimal mean squared error, the optimal agreement, or the conditional entropy. All these measures can be related to one another, and we focused on the conditional entropy (called simply the SBM entropy), as it allows to use tools from information theory. The SBM entropy is defined by the limit (if it exists):

$$\mathscr{H}(a,b) := \lim_{n\to\infty} \frac{1}{n} H(X|G), \tag{1.2}$$

where $H(X|G)$ denotes the conditional entropy of X given G (see Appendix A.1 the definition of conditional entropy). Informally, it measures how much uncertainty is left about the communities after observing the graph.

In Chapter 2 we explain how the characterization of the SBM entropy can be reduced to the computation of entropies on a tree-like graph (i.e. a graph without loops). These tree-entropies can then be efficiently evaluated through belief propagation (BP) (see e.g. Zdeborová and Krzakala, 2016). As a first step, we introduce a "survey" (or side information) ω^{ϵ} that for each node reveals the correct label with probability $1-\epsilon$, and an erasure symbol otherwise. Then, using an interpolation trick, we write the global entropy in (1.2) as an integral of *local* entropies in the SBM model with survey. Few technical steps (see Chapter 2, Section 2.4) allow to write (1.2) in terms of tree-entropies if the following holds:

$$\lim_{k\to\infty} H(\sigma_\rho|\sigma_{L_k}, \omega^{\epsilon}_{T_k}) = \lim_{k\to\infty} H(\sigma_\rho|\omega^{\epsilon}_{T_k}), \tag{1.3}$$

where $\sigma_\rho, \sigma_{L_k}$ are the communities of respectively the root and of the leaves at depth k generated by a broadcasting process on a Poisson tree (see Section 2.4) and $\omega^{\epsilon}_{T_k}$ is a survey that reveals the label of each node with probability $1-\epsilon$. The property in (1.3) implies that conditioned on the survey, the leaves at large depth are irrelevant for retrieving the community of the root node. Thus, we name it *boundary irrelevance (BI)* property. We refer to Chapter 2 for details, formal definitions and results.

1.1.3 Synchronization Models with Survey and Boundary Irrelevance

Motivated by the relevance of the boundary irrelevance property in the characterization of the SBM entropy, we further analyze it on other graphs models. In particular, on a graph with n vertices, we assign a spin (i.e. a value in $\{\pm 1\}$) to each vertex of the graph independently at random. Then, for each edge of the graph we define the edge spin as the product of the spins of its two endpoints, flipped with some probability δ. We assume that a survey reveals the spins of a random fraction of the nodes. We denote X the vertex spins, Y the edge spins and ω^{ϵ} the survey (with ϵ being the fraction of non-revealed nodes). This model is called *synchronization on graphs with side-information*; we refer to Section 2.1 for the formal definition. This definition, or slight generalizations, can capture popular models, such as broadcasting on trees (Evans et al., 2000), censored block models (Heimlicher et al., 2012), stochastic block models.

Previous works (e.g. Abbe and Boix-Adserà, 2018; Abbe et al., 2018; Polyanskiy and Wu, 2018) focused on understanding whether it is information-theoretically possible (or impossible) to have a non-trivial reconstruction of the spins X based on the edge observations Y, depending on the noise of the edge observations and on the graph topology. We consider a similar problem, but with the addition of the survey. Specifically, we analyse whether the revelation of some nodes variables, in addition to the edges observations, makes a difference in the reconstructability of the X, compared to the setting where only the Y are observed. This connects to analysing a boundary irrelevance property, similar to (1.3). We refer to Section 2.1 for definitions and results.

1.2 Neural Networks

The current growth of Machine Learning (ML) research is mainly driven by the success of modern *deep neural networks (DNNs)*. The history of artificial neural networks (ANNs) is believed to have begun in the 40's, when (McCulloch and Pitts, 1943) created the first computational model for neural networks. In the subsequent years, several models inspired by biological learning were introduced (Hebb, 1949; Ivakhnenko and Lapa, 1965; Rosenblatt, 1958). The raise of two major issues (the impossibility to describe the XOR function with the perceptron and the lack of sufficient processing power of the computers of that time) (Minsky and Papert, 1969) lead to a certain skepticism towards ANNs and a subsequent stagnation of research. Interest in ANNs was recovered thanks to the formalization of the backpropagation algorithm (Werbos, 1994), that allowed for the training of multi-layer networks in practice, and to the increase in the availability of computing power, through the use of GPUs and distributed computing. In (Fukushima, 1988), the first networks with weight-sharing appeared. This led to the modern Convolutional Neural Networks (CNNs) (Ciresan et al., 2011), that were shown to beat state-of-the-art performance in image classification tasks. Between 2009 and 2011, new architectures such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) won several competitions in pattern recognition,

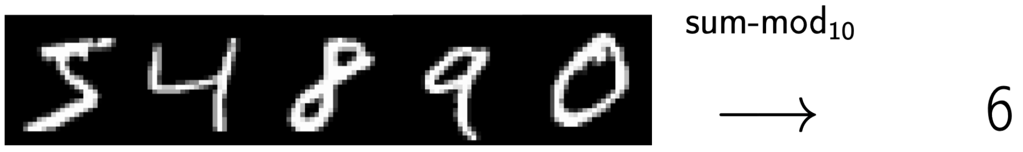

Figure 1.2: Sum modulus-10 of MNIST digits. The inputs are images of $28 \times (28 \cdot 5)$ pixels, consisting of arrays of five MNIST digits. The label is given by the sum mod-10 of the values of the digits, which is equal to 6 in this example.

handwriting recognition, traffic sign recognition, beating human performance. The rise of DNNs marked the beginning of a new era in machine learning, revolutionizing everyday technologies and transforming several fields such as computer vision, natural language processing, product recommendation systems, and financial pricing.

While DNNs have achieved unprecedented accuracy and performance in various tasks, their construction is largely based on trial-and-error heuristics and lacks of a theoretical framework. In the absence of a proper theoretical framework, DNNs are often treated as black-box algorithms, which can lead to a lack of reliability and efficiency in their application.This motivated a vast research effort devoted to the theoretical understanding of the inner mechanisms that make neural networks work. Despite significant progresses, our understanding is limited to fairly restricted settings and in most scenarios used in practice there are several unanswered questions.

1.2.1 Boolean Tasks

While it has been known for a decade that neural networks perform well at image recognition on standard datasets (e.g. on MNIST or CIFAR datasets), their performance on tasks that go beyond classical image recognition is not completely understood. As an example, consider the task pictured in Figure 1.2. The inputs are images of $28 \times (28 \cdot 5)$ pixels, consisting of arrays of five MNIST digits. The label is given by the sum modulus-10 of the values given by the digits. The task for the learner is thus two-fold. Firstly, it has to recognize the MNIST digits, and secondly, it has to learn the logical operation that combines the digits to produce the label, i.e. the sum modulus 10. Another similar example is the Pointer Value Retrieval (PVR), introduced in (Zhang et al., 2021) (Figure 1.3). In the PVR task, the inputs are arrays of MNIST images. The labels are defined by the following procedure. The first digit on the left defines the pointer, and its value indicates the position of a window of a given length, say 2 in the example in Figure 1.3. The label is produced by applying some fixed function (the ‘aggregation’ function) on this window. In order to learn the PVR task, the learner has to first identify the MNIST digits, and secondly, it needs to recognize the logical component, which consists of both the pointer rule and the aggregation function. The PVR benchmark is introduced to investigate the trade-off between memorization and reasoning, by acting with a distribution shift at testing (see Chapter 4 for further analysis of this task). Other examples

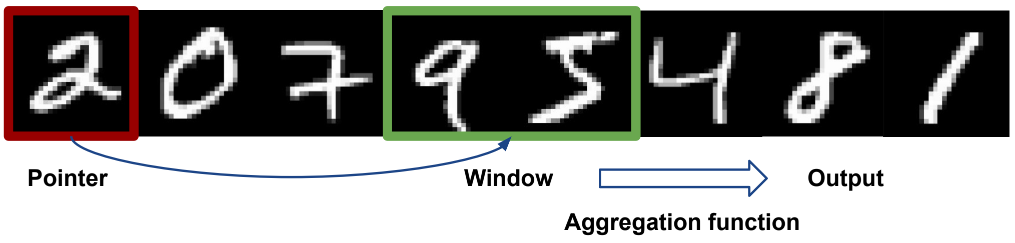

Figure 1.3: Pointer-Value-Retrieval (PVR) task. The inputs are arrays of MNIST images. To define the label: the first digit acts as a pointer, and its value indicates the position of a window of a given length, say 2 in this example; The label is produced by applying an 'aggregation' function on this window.

of tasks with a logical or combinatorial component include, among others, algorithmic datasets (Power et al., 2022) and visual reasoning (Johnson et al., 2017).

These tasks can be modeled by *Boolean functions* that operate on a symbolic representation of the data. The class of Boolean functions includes any function that maps binary bit vectors to real numbers. In the examples mentioned above, we can assign a bit vector to each array of digits by replacing the MNIST images with the values of the corresponding digits in binary alphabet. For instance, the digit '5' can be replaced by '$(0,1,0,1)$', and the digit '9' can be replaced by '$(1,0,0,1)$'. The logical components of the tasks can then be defined by Boolean functions that act on these symbolic representations. To simplify the connection to Boolean analysis, we use binary representations with elements in $\{\pm 1\}$ instead of $\{0,1\}$, as used in e.g. (O'Donnell, 2014). We refer to this mapping of images to binary bit vectors as the "perception" task, and the logical map from binary bit vectors to labels as the "reasoning" task. In this thesis, we assume that the perception is given (i.e. we assume that the network has access to a symbolic representation of the data), and we focus on the reasoning mapping. However, exploring the interplay between perception and reasoning is an interesting direction for future work. We describe our stylized setting in the next Section.

1.2.2 Setting

We assume that the network observes samples of the form $(x, f(x))$, where $x \in \{\pm 1\}^d$ is a d-dimensional binary array and $f : \{\pm 1\}^d \to \mathbb{R}$ is an unknown Boolean target function. We assume that the x are drawn i.i.d. from a distribution $\mathcal{D}^{\text{train}}$. Our goal is to learn the target function with a neural network trained by gradient descent methods.

Fully Connected Networks. Most of this thesis focuses on fully connected neural networks (Figure 1.4). The standard feed-forward fully connected architecture is composed by L layers of neurons, where layer 0 is named *input layer*, layer L is named *output layer* and

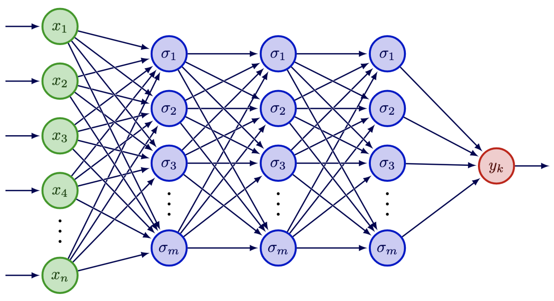

Figure 1.4: Illustration of a fully connected architecture.

layers $1,\dots,L-1$ are called *hidden layers*. Each layer $l \in \{0,\dots,L\}$ has N_l neurons, where in our case $N_0 = d$ (the input dimension) and $N_L = 1$ (the output dimension). A network is defined by L *weight matrices*, denoted by $W^{(l)} \in \mathbb{R}^{N_l \times N_{l-1}}$, for $l = 1,\dots,L$, and L *bias vectors*, denoted by $b^{(l)} \in \mathbb{R}^{N_l}$, for $l = 1,\dots,L$. We denote the parameters of the network by $\theta = (W^{(1)}, b^{(1)},\dots, W^{(L)}, b^{(L)})$. For an input x, we denote by $x^{(l)} = x^{(l)}(\theta)$ the vector containing the outputs of all neurons at layer l. $x^{(l)}$ is defined from the outputs of neurons at previous layers, in the following way:

$$x^{(0)} = x \tag{1.4}$$

$$x^{(l)} = \vec{\sigma}\left(W^{(l)}x^{(l-1)} + b^{(l)}\right), \qquad l = 1,\dots,L-1, \tag{1.5}$$

$$x^{(L)} = W^{(L)}x^{(L-1)} + b^{(L)}, \tag{1.6}$$

where by $\vec{\sigma}$ we denote the function that applies a fixed real function $\sigma : \mathbb{R} \to \mathbb{R}$ component-wise to the input. σ is called the *activation function*. Common choices of activation functions include the *Rectified Linear Unit* ($\mathrm{ReLU}(x) = \max\{0, x\}$), the *sigmoid* ($\sigma(x) = 1/(1+e^{-x})$) or the *squared* function ($\sigma(x) = x^2$). We denote by P the number of parameters in the network.

One well-known result is that neural networks are universal approximators, meaning that given any target function f there exists a neural network with parameters θ^* that can approximate f with high accuracy. Moreover, θ^* can be estimated from data, if enough samples are available. The process of finding the parameters θ^* that can approximate the target function accurately based on data is usually referred to as *training*. The dataset used for training is called *training set*. The goal is to find parameters that minimize the *training loss*, which measure the performance of the network on the training data:

$$\theta^* \in \mathrm{argmin}_\theta \frac{1}{m}\sum_{s=1}^{m} L(f, \theta, x_s), \tag{1.7}$$

where m is the size of the training set and L is a loss function, which is assumed to be differentiable almost everywhere. In this thesis, we will consider the following loss functions:

- Squared loss: $L(f,\theta,x) = (f(x) - \mathrm{NN}(x;\theta))^2$;

- Hinge loss: $L(f,\theta,x) = \max\{0, 1 - f(x)\mathrm{NN}(x;\theta)\}$;
- Correlation loss: $L(f,\theta,x) = -f(x)\mathrm{NN}(x;\theta)$;

The problem defined in (1.7) is called *Empirical Risk Minimization (ERM).* Implementing ERM is difficult for two main reasons: i) high dimensionality: the number of possible models increases exponentially with the number of features, making it computationally hard to search for the optimal model; ii) non-convex optimization: (1.7) is often non-convex and there can be multiple local minima and saddles in the loss function, making it hard to find.

Gradient Descent Algorithms. The most popular algorithm used for training deep neural networks is called *Stochastic Gradient Descent (SGD)*. SGD is an iterative algorithm, that starts from a random assignment of the network parameters, and at each step it computes the gradients of the loss function, using a random subset of the training data (the *mini-batch* or *batch*). It then updates the parameters following the direction of the negative gradient, scaled by a *learning rate.* The learning rate controls the size of the step taken in each iteration, and can be adjusted during training. The updates of SGD are thus defined as follows:

$$\begin{cases} \theta^0 \sim P_0 \\ \theta^{t+1} = \theta^t - \gamma_t \frac{1}{B}\sum_{s=1}^B \nabla_{\theta^t} L(f,\theta^t,x_s), \qquad t \le T, \end{cases} \tag{1.8}$$

where P_0 is an *initial distribution* (usually assumed to be iid Gaussian, or uniform) from which the initial parameters are sampled; γ_t is the learning rate; B is the *batch size.* When the batch size corresponds to the size of the total training set, (5.1) is called *Gradient Descent (GD).* In other words, the stochasticity of SGD is given by the random choice of the mini-batch used to compute the gradients. We consider SGD with *fresh batches*, i.e. at each step, the samples x_s are drawn i.i.d. from $\mathcal{D}^{\text{train}}$.

Furthermore, we consider a slight variation of SGD, called *Noisy-SGD*, where at each step some random noise is injected in the gradients. This variation is commonly considered in statistical query algorithms (Blum et al., 1994; Kearns, 1998) and GD learning (Abbe, Kamath, et al., 2021; Abbe and Sandon, 2020b; Malach et al., 2021). The Noisy-SGD algorithm is defined by the following iterations:

$$\begin{cases} \theta^0 \sim P_0 \\ \theta^{t+1} = \theta^t - \gamma_t\Big(\Big[\frac{1}{B}\sum_{s=1}^B \nabla_{\theta^t} L(f,\theta^t,x_s)\Big]_A + Z^t\Big), \qquad t \le T, \end{cases} \tag{1.9}$$

where Z^t are iid $\mathcal{N}(0,\tau^2\mathbb{I})$, for some *noise level* τ, and are independent from other variables, A is the *gradient range* and by $[.]_A$ we mean that whenever the argument is exceeding A (resp. $-A$), it is rounded to A (resp. $-A$), and the other variables are as defined before. We call A/τ the *gradient precision.* We will assume the gradient precision to be polynomially bounded in the input dimension. When the batch size corresponds to the size of the training set, (1.9) is called *Noisy-GD.*

Generalization Error. While the performance on the training set is used to select the model θ^*, the goal in deep learning is to build a model that can make accurate predictions on new data, and not just the data it was trained on. The ability of the model to generalize to new, unseen data is measured by the *generalization error*. This is defined as:

$$\mathrm{gen}(f,\theta^T) := \mathbb{E}_{x\sim\mathscr{D}^{\mathrm{test}}}\left[L(f,\theta^T,x)\right], \tag{1.10}$$

where $\mathscr{D}^{\mathrm{test}}$ is the *test distribution* and θ^T are the network parameters obtained after training. In words, the generalization error is the expected error of a new data point sampled from a distribution, the test distribution, that is assumed to approximate well reality.

We will consider two settings:

1. *Matched setting:* $\mathscr{D}^{\mathrm{train}} = \mathscr{D}^{\mathrm{test}}$. This is the classical learning setting, where it is asked whether the network can generalize well on a new data point that is similar to the data used for training. We will consider this setting in Chapter 3.
2. *Mismatched setting:* $\mathscr{D}^{\mathrm{train}} \neq \mathscr{D}^{\mathrm{test}}$. In this setting, it is asked whether the network can generalize on data points that are not necessarily similar to the ones observed during training. In Chapter 4, we will consider a setting where the support of the train distribution is a subset of the support of the test distribution, i.e. some samples are withhold during training (*holdout setting*). This can be used to access the trade-off between *memorization* and *reasoning* in neural networks.

1.2.3 Complexity Measures

In this thesis, we will narrow our focus to a specific question:

> Can we characterize the complexity measures that come into play when learning with (S)GD on neural networks?

To analyze the complexity of tasks in this specific algorithmic framework, we seek to identify measures that resemble the VC-dimension (Vapnik, 1999) in the PAC learning framework or the statistical dimension (Blum et al., 2003) in the statistical query (SQ) framework. In particular, we aim to have measures that control the following quantities:

- *Statistical complexity:* the total number of samples used during training. In our setting of SGD with fresh batches, this is given by $B \cdot T$, i.e. the product between the number of samples used at each step (the batch size) and total number of steps.
- *Computational complexity:* the total number of computations performed during training. For SGD, this is controlled by the total number of gradients that are computed during training, which is given by $P \cdot B \cdot T$, i.e. the product of the number of parameters

in the network, the batch size, and the total number of steps. On the other hand, Noisy-GD can be seen as a statistical query (SQ) algorithm[II], and its complexity depends on the number of queries made ($P \cdot T$) and the desired accuracy of each query ($1/\tau$).

- *Generalization performance:* the generalization error achieved after training, as defined in (1.10).

In this thesis, we primarily concentrate on the measures for the computational complexity and generalization performance of neural networks. Below, we provide a list of the measures that we have considered.

1.2.4 Complexity Measures in the Matched Setting

In the following, we denote $\mathscr{D} := \mathscr{D}^{\text{train}} = \mathscr{D}^{\text{test}}$.

Initial Alignment (INAL). In Chapter 3, we pose the following question:

> Is it necessary for a neural network to have some initial alignment with a target function in order for Gradient Descent to be effective over a reasonable time frame? Or could a sufficiently large neural network discover its own correlations with the target even if it is not well-designed from the outset?

To answer this question, and to quantify the answer, we introduce the notion of Initial Alignment (INAL) between a neural network and a target function f. The INAL is defined as the maximum average correlation between f and any neuron in the network:

$$\mathrm{INAL}(f, \mathrm{NN}) := \max_{v \in V_{\mathrm{NN}}} \mathbb{E}_{\theta^0 \sim P_0} \left[\mathbb{E}_{x \sim \mathscr{D}} [f(x) \mathrm{NN}^{(v)}(x; \theta^0)]^2 \right], \tag{1.11}$$

where V_{NN} denotes the set of neurons in the network, and $\mathrm{NN}^{(v)}(x;\theta^0)$ denotes the post-activation output of neuron v at initialization. While (1.11) is applicable to arbitrary input distributions and settings, in the theoretical results in Section 3.1 we focus on the restricted setting of Boolean target functions with inputs uniformly distributed on the hypercube (i.e. $\mathscr{D} = \mathrm{Unif}\{\pm 1\}^d$, with d being the input dimension). We prove that if $\mathrm{INAL}(f, \mathrm{NN})$ is small, which means that at initialization no neuron in the network has sufficient correlation with the target task, then the noisy-GD algorithm (as defined in (1.9)) will not be able to learn f efficiently (Theorem 4). This result concerns the weakest form of learning: we show that if the INAL is small, the network will not be able to achieve accuracy better than guessing with f. This implies that even when the gradients are non-negligible and all weights undergo

[II]Statistical query (SQ) algorithms are algorithms that access data through statistical queries rather than raw data access Kearns, 1998 movement during training, the GD trajectory will remain within regions of the parameter space that do not exhibit correlation with the target function.

Our proof technique holds for any fully connected networks with i.i.d. Gaussian initialization, although our experiments suggest that the INAL may be meaningful for more general architectures and datasets. The Boolean setting allows to make extensive use of the Fourier-Walsh transform (see Section 3.1), which simplifies the analysis, while still capturing meaningful phenomena. We nonetheless expect that the footprints of the proofs derived in Section 3.1 will apply to inputs that are i.i.d. Gaussians or spherical, using different basis than the Fourier-Walsh one. The proof consists of showing that learning a target with low INAL implies learning a function class (the orbit of f) with large statistical dimension (Blum et al., 1994), which is hard for any statistical query (SQ) algorithm (Blum et al., 1994; Kearns, 1998), and in particular for GD. the proof strategy described above holds for any fully connected architectures of any width/depth, and does not require tracking the dynamics of GD, as it relies on SQ arguments. We refer to Section 3.1 for formal definitions and results.

Initial Gradient Alignment. The proof strategy described above presents three main limitations. 1) The argument is only applicable to uniform Boolean inputs. 2) The bound is independent of the network's initialization scale and architecture, and depends solely on the complexity of the target task's orbit and the network's number of parameters (rather than its initialization scale, depth, or width). 3) The argument does not hold for certain 'extreme' functions, such as the parity function on all bits, also known as the *full parity*, defined as $\chi_{[d]}(x) := \prod_{i=1}^{d} x_i$, with $x \sim \{\pm 1\}^d$. In Section 3.2, we address these points by narrowing our analysis to ReLU networks of finite depth trained by noisy-GD on the correlation loss. We consider the alignment between the target and the initial *gradients*, instead of the network's neurons, defined by:

$$G_f(\theta^0) := \mathbb{E}_{x\sim\mathcal{D}}\big[f(x)\nabla_{\theta^0}\mathrm{NN}(x;\theta^0)\big]. \tag{1.12}$$

For i.i.d. Gaussian initialization, we provide a lower bound to the generalization error based on the ℓ_2 norm of $G_f(\theta^0)$, without relying on SQ arguments (Theorem 6). The proof holds for many input distributions, including Gaussian, spherical, non-uniform Boolean distributions. Moreover, the bound depends on the depth of the network and on the initialization scale. As an application, we show that all high degree Boolean functions (including the full parities) are not learnable by a 2-layers fully-connected ReLU network with i.i.d Gaussian initialization trained with noisy-GD on the correlation loss (Theorem 5). Furthermore, for non-Gaussian initializations, we prove a lower bound that depends on the ℓ_4 norm of $G_f(\theta^0)$ (Proposition 7). This is valid for all initial distributions that satisfy a technical assumption, including multivariate Gaussian, uniform. We refer to Section 3.2 for further details.

Noise Stability (Stab). We further consider a classical Boolean measure: namely the *noise-stability* of the target function (O'Donnell, 2014). For a Boolean function f, for a parameter $\delta \in [0,1]$, the $\mathrm{Stab}_\delta(f)$ is defined as (see Def. 14):

$$\mathrm{Stab}_\delta(f) := \mathbb{E}_{x,y}\left[f(x)\cdot f(y)\right], \tag{1.13}$$

where $x \sim \mathrm{Unif}\{\pm 1\}^d$ and y is obtained from x by flipping each coordinate independently with probability δ. $\mathrm{Stab}_\delta(f)$ is a measure of how stable f is to a uniform perturbation of the input coordinates. In Section 3.3, we show that noisy-GD cannot learn tasks that are very sensitive to perturbations of the input coordinates, proving thus a lower bound on the generalization error depending on $\mathrm{Stab}_\delta(f)$ (Theorem 8). This supports a conjecture made in (Zhang et al., 2021). Our proof holds for any fully connected networks with initialization that is invariant to permutation of the input neurons.

1.2.5 Complexity Measures in the Mis-Matched Setting

Boolean Influence (Inf). The Boolean influence comes into play in the mismatched setting, i.e. when train and test distribution are not equal. Our motivating question is the following:

> If we conceal certain samples during the training process, how does this affect the generalization ability of GD? Can GD effectively generalize to unseen domains? Is there a way to quantify its performance in doing so?

We consider a specific distribution-shift setting, named *canonical holdout*, defined as follows. During training one coordinate of the input (say x_k) is frozen to $+1$ and the other coordinates are sampled uniformly and independently in $\{\pm 1\}$. In order words, at training the algorithm observes only half of the hypercube, and it never observes samples where $x_k = -1$. The network is then tested on the full uniform distribution on the hypercube. Thus, $\mathscr{D}^{\mathrm{train}}$ is the uniform distribution on half of the hypercube (with $x_k = 1$) and $\mathscr{D}^{\mathrm{test}} = \mathrm{Unif}\{\pm 1\}^d$. We consider training a neural network with SGD on the squared loss. We observe that the generalisation error admits a tight characterisation in terms of another classical Boolean measure: the Boolean influence of coordinate k on f ($\mathrm{Inf}_k(f)$). Intuitively, $\mathrm{Inf}_k(f)$ measures the importance of coordinate k in f. For binary classification tasks $f : \{\pm 1\}^d \to \{\pm 1\}$, $\mathrm{Inf}_k(f)$ is defined as the probability that flipping x_k changes the output of f. For real-valued f, Inf_k is defined in terms of the Fourier-Walsh transform of f (see Def. 20). We prove the Boolean influence characterisation for linear models (Theorem 9) and we support it experimentally on other architectures, such as MLPs and Transformers (Section 4.5.2). In particular, this puts forward the hypothesis that for such architectures and for learning logical functions, GD tends to have an implicit bias towards low-degree representations. Indeed, we show that the Boolean influence arises when the network learns the lowest degree interpolator of the training data (Lemma 2). The answer to the question above seems to be thus negative: at least in the setting and for the architectures considered, GD seems to not be able to learn the correct task, but instead it favors the low degree representation. An interesting future direction is the following: can we promote other forms of implicit bias (e.g., bias towards symmetrical solutions or minimum description length (MDL) interpolator) that enables to achieve lower generalization error than the Boolean influence?

1.2.6 Curriculum Learning for Parities

The measures mentioned in Section 1.2.4 enable a good understanding of the target functions that are hard to learn through GD on fully connected networks, in the standard matched setting with uniform inputs.

In Chapter 5 we focus on a subset of functions that are known to pose computational barriers, namely the k-parities over d bits of a binary string. This class includes all functions

$$\chi_S(x) := \prod_{i \in S} x_i, \qquad x \in \{\pm 1\}^d, \tag{1.14}$$

such that $S \subseteq [d], |S| = k$. (Abbe and Sandon, 2020a) showed that with uniform inputs the Noisy-SGD algorithm (as defined in (1.9)) with large batch size and low gradient precision cannot learn k-parities in less than $d^{\Omega(k)}$ time. However, if we look at different product distributions (that are not the uniform distribution), learning parities is actually easy, as shown in (Daniely and Malach, 2020; Malach et al., 2021). For instance, if the inputs are generated such that $x_i \overset{\text{iid}}{\sim} \text{Rad}(p)$ [III] for $i = 1, ..., d$, with $p = 1 - 1/k$, computing correlations with each bit will recover the parity in time linear in d and polynomial in k. However, it turns out (see experiments in Section 5.3.2) that training on examples sampled from a biased measure alone is not sufficient to learn the parity under the unbiased measure, with SGD on fully connected networks. This is because the training set sampled under $\text{Rad}(p)^{\otimes d}$ has essentially no examples with Hamming weight [IV] close to $d/2$, due to concentration of Hamming weight. Therefore, it is not reasonable to expect a general algorithm like gradient descent on a fully connected network (which does not know that the target function is a parity) to learn the value of the function on such inputs. However, training on the biased measure can help us identify the support of the parity faster than if we trained under the uniform distribution.

We consider a training strategy that consists of generating and presenting samples in a meaningful order: We initially train on inputs sampled from $\text{Rad}(p)^{\otimes d}$, with p close to 1, then we move (either gradually or by sharp steps) towards the unbiased distribution. We call this a *Curriculum Learning (CL)* strategy for learning k-parities. We show that with this training strategy, we can learn a target k-parity with a computational cost of of $d^{O(1)}$ with a fully connected network trained with SGD on the hinge loss or on the covariance loss (see Def. 25). Our results are valid for any (even) k and d, thus for large k and d our bound is

[III] $x_i \sim \text{Rad}(p)$ if $\mathbb{P}(x_i = 1) = 1 - \mathbb{P}(x_i = -1) = p$.
[IV] The Hamming weight of $x \in \{\pm 1\}^d$ is $H(x) = \sum_{i=1}^d \mathbb{1}(x_i = 1)$.

below the $d^{\Omega(k)}$ lower bound for bounded gradient precision models under the uniform distribution.

Furthermore, we show that for another class of functions - namely the *'Hamming mixtures'* - CL strategies involving a bounded number of product distributions are not beneficial, while we conjecture that CL with unbounded many product distributions can learn this class efficiently.

1.3 Outline of the Thesis

This thesis is divided into two parts.

Part I concerns the block models. In Chapter 2 we introduce the property of boundary irrelevance in spin synchronization models, we present our results on tree- graphs and we show how such property can be used to characterize the SBM entropy. Part of Chapter 2 is based on (Abbe, Cornacchia, et al., 2021).

Part II concerns neural networks. In Chapter 3, we introduce and analyse the complexity measures that come into play in the matched setting (i.e. when $\mathscr{D}_{\text{train}} = \mathscr{D}_{\text{test}}$): the Initial Alignment, the Initial Gradient Alignment, and the Noise Stability. We show how the generalization error achieved after training with noisy-GD on neural networks can be lower bounded in terms of these measures. This Chapter is based on (Abbe, Cornacchia, et al., 2022), part of (Abbe, Bengio, et al., 2022), and on a work in preparation (Abbe and Cornacchia, 2023). In Chapter 4, we analyse the interplay between the Boolean influence and the generalization error in a specific mis-matched setting (i.e. when $\mathscr{D}_{\text{train}} \neq \mathscr{D}_{\text{test}}$). This Chapter is based on part of (Abbe, Bengio, et al., 2022). In Chapter 5 we analyse a 'curriculum learning' strategy for learning k-parities. This Chapter is based on (Cornacchia and Mossel, 2023).

Block Models Part I

2 Block Models with Side Information

2.1 Spin Synchronization on Graphs with Side-Information

Spin synchronization models have been considered in several works, such as (Abbe and Boix-Adserà, 2018; Abbe et al., 2018; Polyanskiy and Wu, 2018). Their attractiveness is in part motivated by their ability to capture crucial models such as broadcasting on trees (Evans et al., 2000), stochastic block models (Abbe, 2017), censored block models (Heimlicher et al., 2012). Here, we consider a slight variant of the classical spin synchronization models where some *side-information* about the vertex variables is available.

Definition 1 (Spin Synchronization on Graphs with Side-Information)**.** *Let $G = ([n], E)$ be a graph with n vertices. Let $X_1, ..., X_n \overset{iid}{\sim} \mathrm{Rad}(1/2)$*[I] *be n independent spins, laying on the vertices of the graph. For every edge $uv \in E$, we observe $Y_{uv} = X_u X_v Z_{uv}$, where $Z_{uv} \overset{iid}{\sim} \mathrm{Rad}(\delta)$, for some δ positive. In addition to the edge observations Y_E, we consider independent observations on the vertices. For every vertex $v \in [n]$, let $\omega_v^\epsilon = X_v$, with probability $1-\epsilon$, and $\omega_v^\epsilon = *$ with probability ϵ.*

We call ω^ϵ *BEC side-information*, where BEC stands for *binary erasure channel* (notice that $\omega_v^\epsilon = \mathrm{BEC}_\epsilon(X_v)$, with ϵ being the erasure probability[II]). This is to distinguish our setting from settings with *BSC side-information*, considered for instance in (Mossel and Xu, 2015; Rebeschini and van Handel, 2015), where for every vertex v, a noisy version of X_v is observed. In the following we will consider BEC side-information, which we will call *side-information* or *survey*, for brevity. We call (X, ω^ϵ, Y) a synchronization instance with survey on G.

Previous works (e.g. Abbe and Boix-Adserà, 2018; Abbe et al., 2018; Polyanskiy and Wu, 2018) focused on understanding whether it is information-theoretically possible (or impossible) to have a non-trivial reconstruction of the spins X based on the edge observations Y_E, depending e.g. on the noise of the edge observations and on the graph topology. Specifically,

[I] $z \sim \mathrm{Rad}(p)$ if $\mathbb{P}(z = 1) = 1 - \mathbb{P}(z = -1) = p$.

[II] We refer to Appendix A.1 for the definition of such channel and for background on information theory.

a non-trivial reconstruction of the spins is possible when the mutual information[III] between the spins of two vertices $u, v \in [n]$ at distant sites, conditioned on Y_E, i.e.

$$I(X_u; X_v | Y_E), \tag{2.1}$$

is non-vanishing as their distance and n grow. In this case, we say that we are in the *reconstruction regime.* On the other hand, if (2.1) is vanishing with n, i.e. if the Y_E are asymptotically uninformative of X, we say that we are in the *non-reconstruction regime.*

We consider a similar problem to the above, but with the addition of the side-information ω^ϵ. Specifically, we analyse whether the revelation of some nodes variables, in addition to the edges observations, makes a difference in the reconstructability of the X, compared to the setting where only the Y_E are observed. In other words, we consider the quantity

$$I(X_u; X_v | \omega^\epsilon, Y_E) \tag{2.2}$$

and we compare it with (2.1). We analyse (2.2) in specific settings. Equivalently to 2.1 and 2.2, one can consider the mutual-information between the spin of an arbitrary vertex o and the spins of all vertices at large distance from o, as we do in Section 2.2.

As mentioned, several inference problems on graphs can be viewed as special cases of spin synchronization models. As examples, we define two models that are of particular interest for us: the stochastic block model (SBM) and the broadcasting on trees (BOT) model.

Example 1 (Stochastic Block Model (SBM))**.** *We define the SBM with two symmetric communities. Let n be a positive integer (the number of vertices). The pair (X, G) is drawn under* $\mathrm{SBM}(n, q_{\mathrm{in}}, q_{\mathrm{out}})$ *if X is an n-dimensional random vector with i.i.d. components distributed under* $\mathrm{Rad}(1/2)$*, and G is an n-vertex simple graph where vertices i and j are connected with probability q_{in} if $X_i = X_j$ and with probability q_{out} if $X_i \neq X_j$, independently of other pairs of vertices.* Community recovery *refers to the problem of detecting the vertex assignments X based on observation of the graph structure G. This can be related to the model in Definition 1 by replacing the edge observations Y_E by indicators of the edges in G (e.g. $Y_{uv} = (-1)^{1-\mathbb{1}(uv \in G)}$)*[IV]*. One can also add a survey that reveals the communities of a fraction of the vertices, as done in e.g. (Kanade et al., 2014).*

Example 2 (Broadcasting on Trees (BOT))**.** *Let T be a rooted tree. Let $\sigma_0 \sim$* $\mathrm{Rad}(1/2)$ *be the root bit and assume that it is broadcast through each edge independently with flip probability η. Let σ_t be the leaves bits at depth t, and $\sigma_{\leq t}$ be the set of vertex bits at depth $\leq t$ from the root. Let $\omega^\epsilon_{\leq t}$ be the output of a "survey" or a "side-information" that reveals each vertex of the tree until depth t (excluding the root) independently with prob $1 - \epsilon$. This model is equivalent to the one in Definition 1, in the sense that for appropriate δ and η, and for any t, $I(X_0; X_t | Y_E, \omega^\epsilon) = I(\sigma_0; \sigma_t | \omega^\epsilon)$ (see for instance Proposition 2.1 in Abbe and Boix-Adserà,*

[III]See Appendix A.1 for the definition of mutual information.

[IV]In this case the channel $\mathbb{P}(Y_{uv} | X_u \cdot X_v)$ is not simply of the type $X_u X_v Z_{uv}$, but one can extend Def. 1 to include more general edges interactions.

2019).

The presence of side-information in graph inference problems is not novel and has several motivations. For example, (Kanade et al., 2014) considers the so-called *labeled* stochastic block model, where the true cluster assignments of some nodes variables are available. There, the presence of side information breaks the symmetry of the unlabeled SBM and, in some regimes, it can affect the detectability thresholds and the fraction of nodes that are correctly clustered. Also, (Mossel and Xu, 2015) introduces a local belief propagation algorithm for the binary symmetric SBM with BSC side-information and proves that it has optimal performance in some regimes (specifically for either very small or very large SNR, or for very accurate side-information). Moreover, (Alaoui and Montanari, 2019) used the BEC side-information for proving the non-existence of an information-computational gap in amenable graphs, and the existence of such a gap in regular trees. From a slightly different perspective, (Rebeschini and van Handel, 2015) analyses the spatial mixing property (i.e. whether the correlation between spins at distant sites is vanishing or not) and the spatial conditionally mixing property (that is the mixing property conditioned on some side-information) of Markov random fields on two dimensional lattices.

2.2 Boundary Irrelevance

We focus on two properties of spin synchronization models with side-information. To illustrate them, let us fix some notation. For a vertex $o \in [n]$, we denote by $B_n(o) = \{u \in [n] : d_G(o,u) \leq n\}$, the set of vertices at distance at most n from o in G (where the distance between two vertices is the length of the shortest paths between them), and by $\partial B_n(o)$ the set of vertices at distance exactly n from o. For (X, ω^ϵ, Y) a synchronization instance with survey, we consider the following two properties:

1. **Boundary Irrelevance in the Non-Reconstruction Regime (BIN)**:

$$\text{If } \lim_{n\to\infty} I(X_o; X_{\partial B_n(o)} | Y_{B_n(o)}) = 0 \implies \lim_{n\to\infty} I(X_o; X_{\partial B_n(o)} | Y_{B_n(o)}, \omega^\epsilon_{B_n(o)}) = 0, \quad \forall \epsilon \in [0,1],$$

   namely a synchronization instance has the (BIN) property if in the standard non-reconstruction regime, the boundary of $B_n(o)$ gives no information about the label of o also in our setting with BEC survey.

2. **Boundary Irrelevance in General Regime (BI)**:

$$\lim_{n\to\infty} I(X_o; X_{\partial B_n(o)} | Y_{B_n(o)}, \omega^\epsilon_{B_n(o)}) = 0, \quad \forall \epsilon \in [0,1),$$

   i.e. $H(X_o | X_{\partial B_n(o)}, Y_{B_n(o)}, \omega^\epsilon_{B_n(o)}) \sim H(X_o | Y_{B_n(o)}, \omega^\epsilon_{B_n(o)})$ in every regime (regardless of the value of the edges noise δ).

Note that the (BI) is a much stronger property that the (BIN). In particular, taking the limit $\epsilon \to 1^-$, the (BI) implies that revealing an (arbitrarily) small fraction of vertex labels gives the same information about the bit of vertex o, as revealing the whole boundary labels at large distance, even in the reconstruction regime.

The rest of this Chapter is structured as follows:

1. In Section 2.3, we show that the (BIN) holds for any tree-like graph.
2. In Section 2.3.1 we show that the (BI) does not hold for a particular type of trees, i.e. the "stringy trees".
3. In Section 2.4 we explain how the (BI) can be applied to characterize the entropy of the stochastic block model (SBM) (see below for definition and context).

We Conjecture that the (BIN) holds for any graph. Some related thoughts will be discussed in Section 2.5.

Conjecture 1. *For every binary synchronization instance with side-information on a general graph G, the (BIN) holds.*

Furthermore, the (BI) has been shown to hold on regular and Poisson trees [V] (Yu and Polyanskiy, 2022). In particular, it has been shown that the fixed points of the belief propagation (BP) recursion in the setting with survey only and in the setting with survey and leaves coincide. In particular, the two settings with survey only and with survey and leaves satisfy the same BP recursion equation, but they have different initialization (corresponding to different information at depth exactly t). In Section 2.4 we explain connection to this property and the SBM entropy.

2.3 Trees

Firstly, we show that the (BIN) holds for every trees. The result is a direct consequence of the following proposition.

Proposition 1. *Let (X, ω^ϵ, Y) be a synchronization instance with survey on a tree T, rooted at 0. Let X_t denote the spins of all vertices at depth t. Then, for every $\epsilon \in [0,1]$ and for every $t \in \mathbb{N}$, $I(X_0; X_t | Y, \omega^\epsilon) \leq I(X_0; X_t | Y)$.*

The proof of Proposition 1 is in Appendix A.2.

Corollary 1. *For every binary synchronization instance with side-information on a tree, the (BIN) holds.*

[V] A Poisson tree, or Galton-Watson tree with Poisson offspring, is a random tree where the offspring of each node is distributed as Poisson(d), for some d.

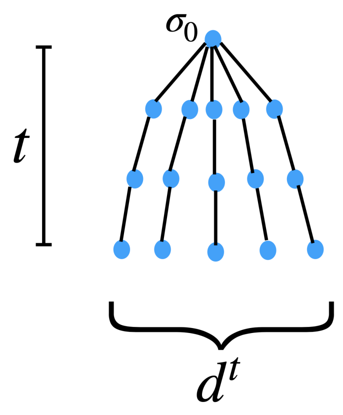

Figure 2.1: Illustration of a ‘stringy tree’.

2.3.1 Stringy Trees.

On the other hand, the (BI) is in general not true for every trees. We show that it does not hold for a specific type of trees: the “stringy-trees”. A stringy tree consists of a root with d^t independent paths from it, each of length t (see Figure 2.1). Consider a broadcasting process on a stringy-tree from the root bit to the leaves at depth t. The (BI) would hold if the conditional distribution of the root bit given the leaves at depth t and the side-information was asymptotically equal to the conditional distribution of the root given the side-information only. In this setting, computing these marginal distributions is feasible since it requires only one step of Bayes rule (or belief propagation). In fact, each branch i gives an independent belief that is the output of a BSC channel [VI] of flip probability $\frac{1}{2}-\frac{1}{2}(1-2\eta)^{T_i}$, where T_i is a random variable defined as follows: In the survey-only setting, T_i denotes the depth of the first surveyed node in branch i if above depth t, and $T_i = \infty$ if the first surveyed node is below depth t; In the survey and leaves setting, T_i denotes the minimum between t and the depth of the first surveyed node. Recalling Example 2, we denote by η the probability that two adjacent vertices have different spins. We get the following Theorem, which implies that if $d(1-2\eta)^2 > 1$, then there exists some $\epsilon \in [0,1)$ for which boundary nodes conditioned on surveyed nodes bring non-negligible information for reconstructing the root bit, hence the (BI) does not hold.

Proposition 2. *Let (X,ω^ϵ,Y) be a synchronization instance on a stringy tree with d^t branches and flip probability η. Then,*

$$\lim_{t\to\infty} I(X_0;X_t|Y,\omega^\epsilon) \begin{cases} =0 & \text{if } \epsilon d(1-2\eta)^2 < 1, \\ >0 & \text{if } \epsilon d(1-2\eta)^2 \geq 1. \end{cases} \tag{2.3}$$

The proof of Proposition 2 is in Appendix A.3.

[VI] For $X \in \{\pm 1\}$, $\mathrm{BSC}_\delta(X) := X \cdot Z_\delta$, where $Z_\delta \sim \mathrm{Rad}(1-\delta)$ and Z_δ is independent of X. We call δ the “flip probability”.

Corollary 2. *For some synchronization instances with survey on stringy trees, the (BI) does not hold.*

2.4 Stochastic Block Model Entropy

Background. Over the last decade, several works have established a precise picture for the statistical and algorithmic behavior of the stochastic block model (see an account in Abbe, 2017). In particular, the questions of weak and exact recovery, i.e., whether it is possible (or not) to recover the communities in the extremal cases of weak and exact accuracy, have been fully closed in the two-community symmetric SBM by establishing sharp threshold phenomena in terms of appropriate signal-to-noise (SNR) ratios (Abbe et al., 2016; Massoulié, 2014; Mossel et al., 2015, 2018). Yet, the more nuanced question of proving how much information or agreement can be recovered about the communities at any given value of the SNR has remained open for several years even in this simplest case.

More specifically, for two symmetric communities and in the sparse regime, the expression of the limiting entropy of the SBM has been characterized[VII] at all SNR for the special case of disassortative communities (i.e., communities that connect more outside than inside) in (Coja-Oghlan et al., 2017). The problem for assortative communities but in the denser regime, where the vertex degrees diverge while maintaining a finite SNR, has been closed in (Deshpande et al., 2016). For the classical case of assortative communities and in the sparse regime, significant progress has been made in (Kanade et al., 2014; Mossel et al., 2016; Mossel and Xu, 2015). The expression of the optimal agreement (rather than the entropy) has been settled in these works for SNR large enough, and is related to the problem of robust reconstruction on a tree (Mossel et al., 2016). The reduction to the problem of boundary irrelevance on trees, discussed below, has been used to characterize the SBM entropy for all SNR outside the interval $(1, 3.513)$ in (Abbe, Cornacchia, et al., 2021), and to close the problem for all SNR in (Yu and Polyanskiy, 2022).

The SBM Entropy. Recall, the SBM model defined in Example 1. In particular, in the symmetric SBM with two communities in the sparse regime, a random variable X is drawn uniformly at random in $\{\pm 1\}^n$ and an n-vertex graph G is drawn by connecting vertices having same (resp. different) values in X with probability a/n (resp. b/n).

The SBM mutual information is defined by the limit (if it exists)

$$\mathscr{I}(a,b) := \lim_{n\to\infty} \frac{1}{n} I(X;G), \tag{2.4}$$

where I denotes the mutual information (see Appendix A.1). Note that establishing the

[VII] Characterizing the limit does not mean obtaining an explicit expression; it refers to an implicit n-independent expression relying on integrals and fixed point equations for the quantities of interest in all of the references discussed here.

existence of this limit is nontrivial. This was proved in ("Conditional Random Fields, Planted Constraint Satisfaction, and Entropy Concentration", 2015) for the case of $a < b$, the same case for which the value of the limit has more recently been established (Coja-Oghlan et al., 2017). Note also that due to the chain rule $I(X;G) = H(X) - H(X|G)$, the SBM mutual information is the complement of the SBM conditional entropy (called simply the SBM entropy):

$$\mathcal{H}(a,b) := \lim_{n\to\infty} \frac{1}{n} H(X|G). \tag{2.5}$$

Informally, the SBM mutual information measures how much information can be recovered about the communities after observing the graph, and equivalently, the SBM entropy measures how much uncertainty is left about the communities after observing the graph. More formally, it quantifies the average number of bits needed to represent the communities after observing the graph.

Note that one may use other measures on the communities signal given the graph, such as the optimal (normalized) mean square error of reconstructing the $n \times n$ rank-2 block matrix (with a/n in the $n/2 \times n/2$ diagonal blocks and b/n in the off diagonal blocks), or the optimal (normalized) agreement (Hamming distance) of reconstructing X up to a community relabelling. These can be explicitly related to each other in the tree models discussed next, and require bounds in the SBM context; see for instance (Deshpande et al., 2016). The conditional entropy allows however for a direct reduction from the SBM to the tree model with side information, as discussed below.

The BOTS Entropy. Consider the problem of broadcasting on a tree with side information (BOTS), defined in Example 2. This can be defined on general trees and with general side information, but consider for simplicity the case of regular trees (where each vertex has exactly d descendants) and erasure side information. In this model, a random bit is attached to the root of the tree and broadcasted down the tree by flipping its value independently with probability δ on each edge (for convenience we call $\theta = 1 - 2\delta$). We denote by σ_ρ the root bit, by σ_{L_k} the d^k-dimensional vector of the leaf bits at generation k, and by $\omega^{\epsilon}_{T_k}$ the side information up to depth k: these are the vertices labelled that are revealed in the tree (besides the root) independently with probability $1-\epsilon$. We call this side information the "survey". Note that this is the type of side information used in our connection between BOTS and SBM entropies, but other types of side information are of independent interest.

We are now interested in two quantities:

1. the limiting entropy of the root bit after observing the leaf bits and the survey, i.e.[VIII],

$$\bar{h}(d,\theta,\epsilon) := \lim_{k\to\infty} H(\sigma_\rho | \sigma_{L_k}, \omega^{\epsilon}_{T_k}),$$

[VIII]In these tree models, the limits can be proved to always exist.

2. the same quantity without the leaf bits being observed, i.e.,

$$h(d,\theta,\epsilon) := \lim_{k\to\infty} H(\sigma_\rho | \omega^\epsilon_{T_k}).$$

We now give a rather direct method to express the SBM entropy in terms of BOTS entropies.

SBM to BOTS Entropy Reduction. The relation obtained between the SBM and BOTS conditional entropy is as follows: if for some range of parameters d, θ, we can establish that

$$h = \bar{h}, \quad \forall \epsilon \in (0,1),$$

i.e., if the (BI) holds, then we can characterize $\mathcal{H}$ as an integral of $\bar{h}$ using the parameter correspondence $d = (a+b)/2$ and $\theta = \frac{a-b}{a+b}$ (see Theorem 1).

Our starting point to such a reduction is an area-theorem or interpolation trick that is commonly used in coding theory (T. Richardson and R. Urbanke, 2001) and related statistical physics literature (Mézard and Montanari, 2009).

The idea is to express the entropy in the SBM $H(X|G)$ as the integral

$$\frac{1}{n}H(X|G) = \int_0^1 \frac{1}{n}\frac{\partial}{\partial \epsilon} H(X|G,\omega^\epsilon)d\epsilon, \tag{2.6}$$

where, similarly as before, ω^ϵ is an erasure survey that reveals the community of each vertex in X independently with probability $1-\epsilon$. We then use the fact that $\frac{1}{n}\frac{\partial}{\partial\epsilon}H(X|G,\omega_\epsilon) = H(X_1|G,\omega^\epsilon_{\sim 1})$, where 1 is an arbitrary vertex in the graph and $\omega^\epsilon_{\sim 1}$ denotes the erasure survey on all vertices excluding vertex 1. Since conditioning reduces entropy, one can upper bound $H(X_1|G,\omega^\epsilon_{\sim 1})$ by considering only the information in the vertex 1 neighborhood, and due to the local tree-like topology of SBMs, this gives an upper bound with the BOTS entropy without leaf information. Moreover, one can add the leaf information in the conditioning to cut-off the graph beyond a local neighborhood, using the Markovianity[IX] of the model, obtaining as well a lower bound from the BOTS entropy but this time with the leaf information, cf. (A.32) in Appendix A.4.

Different kind of reductions from SBMs to tree models have long been known and leveraged in the SBM e.g. in (Alaoui and Montanari, 2019; Coja-Oghlan et al., 2017; Mossel et al., 2016).

For our characterization to hold, we need to establish $h = \bar{h}$.

[IX]Strict Markovanity does not hold in the SBM due to the weak effect of non-edges, and this requires a technical lemma; see proof of Theorem 1. This technicality can also be avoided by considering the related Censored Block Model (CBM), rather than the SBM, for which strict Markovianity holds.

Uniqueness of BP Fixed Point for BOTS. Establishing $h = \bar{h}$ is equivalent to establishing that the BOTS associated distributional fixed point equation (known as BP fixed point) has a unique solution. This automatically has several implications.

First, this establishes the desired "Boundary Irrelevance" (BI) property for BEC survey, i.e., $h = \bar{h}$:

$$\lim_{k\to\infty} H(\sigma_\rho|\sigma_{L_k}, \omega^\epsilon_{T_k}) = \lim_{k\to\infty} H(\sigma_\rho|\omega^\epsilon_{T_k}). \tag{2.7}$$

This implies

$$\lim_{\epsilon\to 1}\lim_{k\to\infty} H(\sigma_\rho|\omega^\epsilon_{T_k}) = \lim_{k\to\infty} H(\sigma_\rho|\sigma_{L_k}). \tag{2.8}$$

Indeed, one only needs to notice that $\lim_{\epsilon\to 1}\lim_{k\to\infty} H(\sigma_\rho|\sigma_{L_k}, \omega^\epsilon_{T_k}) = \sup_{\epsilon,k} H(\sigma_\rho|\sigma_{L_k}, \omega^\epsilon_{T_k})$ and that for every k the latter quantity is continuous in $\epsilon \in [0,1]$ including at the boundary.

Further, the presence of the survey allows to convert the absence of leaf information into the presence of noisy leaf information, thereby obtaining the robust reconstruction property in the presence and in the absence of the survey (Mossel et al., 2016).

Property (2.8) is also known in the SBM literature as the condition for "optimality of local algorithms", and was investigated in (Kanade et al., 2014; Mossel and Xu, 2015). These works build on the crucial contribution of (Mossel et al., 2016), which shows uniqueness of BP fixed point for BOT without survey and $d\theta^2 > C$, where C is "large enough". Note that since the conditional entropy in (2.8) can be sandwiched between $H(\sigma_\rho|\sigma_{L_k})$ and $H(\sigma_\rho|\omega^\epsilon_{L_k})$, the result of (Mossel et al., 2016) implies (2.8), as indeed observed in (Kanade et al., 2014, Prop. 3).

2.4.1 Formal Results: Boundary Irrelevance and SBM Entropy

In this Section, we present the formal characterization of the SBM entropy that appeared in (Abbe, Cornacchia, et al., 2021) for all SNR $\notin (1, 3.513)$. Let us first redefine formally the broadcasting on trees model with survey, introduced in Example 2.

Definition 2 (Broadcasting on Trees with Survey (BOTS))**.** *We start with the standard broadcasting on trees (BOT) setting. Let T be an infinite tree rooted at ρ. Let $\sigma_\rho \sim$* Unif$(\{\pm 1\})$ *be the root bit and assume that it is broadcast through each edge independently with flip probability $\delta \in (0, \frac{1}{2}]$. For simplicity we use notation $\theta = 1 - 2\delta$. Let L_k denote the set of nodes at level k, and T_k denote the set of nodes at level $\leq k$ (where the root is at level 0). Reconstruction on such models consists of recovering the root bit after observing the leaves bits at large depth (Evans et al., 2000). We consider a slightly different problem, where we have access to some node side-information, or "survey". Specifically, let ω^ϵ be a survey that for each node reveals the correct label with probability* $1-\epsilon$ *and an erasure symbol otherwise. We call (T, ρ, θ) a broadcasting instance with survey.*

We call the specific survey above the "erasure" survey, which appears in (Kanade et al., 2014).

One can replace the erasure survey by other channels, and obtain for instance the setting considered in (Mossel and Xu, 2015), where for each node u, $\mathbb{P}[\omega_u = \sigma_u] = 1 - \mathbb{P}[\omega_u = -\sigma_u] = 1 - \alpha$. The erasure survey is of particular interest to us because of its application to the computation of the SBM entropy (Theorem 1).

Theorem 1. *Let* $(X, G) \sim SBM(n, 2, a/n, b/n)$. *Let* T *be a Galton-Watson tree with Pois*$(\frac{a+b}{2})$ *offspring distribution and let* $(T, \rho, \frac{a-b}{a+b})$ *be a broadcasting instance with erasure survey, and edge flip probability* $\frac{b}{a+b}$. *Let* $\alpha^* \approx 3.513$ *be the unique solution in* $\mathbb{R}_{>1}$ *to the equation* $\exp(-\frac{\alpha-1}{2})\alpha = 1$. *The following hold. For* a, b *such that* $\frac{(a-b)^2}{2(a+b)} \leq 1$ *or* $\frac{(a-b)^2}{2(a+b)} \geq \alpha^* \approx 3.513$

$$\mathscr{H}(a,b) = \lim_{n\to\infty} \frac{1}{n} H(X|G) = \int_0^1 \lim_{k\to\infty} H(\sigma_\rho | T, \sigma_{L_k}, \omega^{\epsilon}_{T_k}) d\epsilon. \tag{2.9}$$

A crucial ingredient to establish Theorem 1 is the Boundary Irrelevance (BI) property for BOTS. We focus on regular and Galton-Watson trees with Poisson offspring.

Theorem 2. *Let* T *be a* d*-regular tree or a Galton-Watson tree with Poisson*(d) *offspring distribution, with root vertex* ρ. *The (BI) holds for any* (T, ρ, θ), *with* $\epsilon \in [0,1)$, *with* $d\theta^2 < 1$ *or* $d\theta^2 > \alpha^*$, *where* $\alpha^* \approx 3.513$ *is the unique solution in* $\mathbb{R}_{>1}$ *to the equation* $\exp(-\frac{\alpha-1}{2})\alpha = 1$.

We remark that Theorem 2 is a relaxation of a stronger Theorem valid for all surveys that are BMS channels. We refer to (Abbe, Cornacchia, et al., 2021) for such stronger result and proof. We refer to (Yu and Polyanskiy, 2022) for extension of Theorem 2 to all values of $d\theta^2$, which closes the characterization of the SBM entropy in the case of two symmetric communities. We refer to Section A.4 for the proof of Theorem 1 from Theorem 2.

2.5 Conclusion and Future Works

We have shown that the (BIN) holds for any trees. The next step would be to prove it for some loopy graphs, and ideally to close Conjecture 1 for any graph. For instance, one can try to generalize the approach that we used for trees to loopy graphs. In particular, if one could show that for any graph G and for any sets $U, V \in V(G)$, it holds that

$$I(X_o; X_S, X_U | Y_G) \leq I(X_o; X_S | Y_G) + I(X_o; X_U | Y_G), \tag{2.10}$$

then the (BIN) would follow. We haven't found a counter-example to (2.10). On the other hand, in the case of 2D-grids, (Rebeschini and van Handel, 2015, Conj. 5.10) Conjectured the (BIN) property for BSC side-information, instead of BEC side-information. However, we do not believe that this extension is trivial. Specifically, the BEC survey allows to know the locations where erasure happened, whereas when using the BSC side-information we have no information about the locations of the flipped bits.

Regarding the (BI), we have shown that it does not hold for stringy trees. Moreover, it has been shown that the (BI) holds for regular and Poisson trees (Abbe, Cornacchia, et al., 2021; Yu and Polyanskiy, 2022). It would be interesting to characterize the set of graphs for which the (BI) holds. One observation is that the stringy trees are a very specific class of graphs, in the sense that the neighborhood of the root changes as t increases (where t is the depth of the leaves that we look at), i.e. the sequence of trees T_t is not locally convergent (cfr. Alaoui and Montanari, 2019, Def. 1). One could then argue that the (BI) hold for every binary synchronization instance with survey on a locally convergent sequence of graphs.

Additionally, it would be interesting to examine more general settings. For instance, one could think of more general definitions of side-information, e.g. revealing interactions of sets of node variables. (Rebeschini and van Handel, 2015) hypothesize that the (BIN) holds in any setting with the absence of observations symmetries. An observation is symmetric if it stays unchanged under a global flip of all labels. For instance, in our setting the edge observations only are symmetric, whereas they are no longer symmetric if we add the side-information. It would be interesting to analyze this problem.

Finally, one could ask whether our results could hold in settings with non-binary labels. The subadditivity property that we used for trees is specific to binary synchronization instances. As for the SBM entropy, while our reduction to boundary irrelevance on trees hold beyond the binary case, proving (or disproving) the (BI) for general alphabet is still open, even in the case of regular and Poisson trees. For ternary alphabets, we are not aware of any counter-example to the (BI). The analysis of the fixed points of the BP recursions for ternary alphabets requires different techniques than the ones used in the binary case, and as such it is not an easy extension. Recently, (Gu and Polyanskiy, 2023) proved the (BI) for large alphabets in some regimes, and they showed that in the case of 4 communities, the (BI) does not hold in all regimes. For regular trees and large alphabets, (Alaoui and Montanari, 2019) showed that below KS, the survey gives no information about the root label, which is in contrast with the (BI), since for non binary alphabets, weak recovery is possible below KS.

Neural Networks Part II

3 Matched Setting

3.1 Initial Alignment

Does one need an educated guess on the type of architecture needed in order for gradient descent to learn certain target functions? Convolutional neural networks (CNNs) have an architecture that is natural for learning functions having to do with image features: at initialization, a CNN is already well posed to pick up correlations with the image content due to its convolutional and pooling layers, and gradient descent (GD) allows to locate and amplify such correlations. However, a CNN may not be the right architecture for non-image based target functions, or even certain image-based functions that are non-classical (Liu et al., 2018). More generally, we raise the following question:

> Is a certain amount of 'initial alignment' needed between a neural network at initialization and a target function in order for GD to learn on a reasonable horizon? Or could a neural net that is not properly designed but large enough find its own path to correlate with the target?

In order to formalize the above question, one needs to define the notion of 'alignment' as well as to quantify the 'certain amount' and 'reasonable horizon' notions. We focus on the 'polynomial-scaling' regime and on fully connected architectures, but we conjecture that a more general quantitative picture can be derived. Before defining the question formally, we stress a few connections to related problems.

A different type of 'gradual' question has recently been investigated for neural networks, namely, the 'depth gradual correlation' hypothesis. This postulates that if a neural network of low depth (e.g., depth 2) cannot learn to a non-trivial accuracy after GD has converged, then an augmentation of the depth to a larger constant will not help in learning (Allen-Zhu and Li, 2020; Malach and Shalev-Shwartz, 2019). In contrast, the question studied here is more of a 'time gradual correlation' hypothesis, saying that if at time zero GD cannot correlate non-trivially with a target function (i.e., if the neural net at time zero does not have an initial alignment), then a polynomial number of GD steps will not help.

From a lower-bound point of view, the question we ask is also slightly different than the traditional lower-bound questions posed in the learning literature that have to do with the difficulties of learning a class of functions irrespective of a specific architecture. For instance, it is known from (Blum et al., 1994; Kearns, 1998) that the larger the statistical dimension of a function class is, the more challenging it is for a statistical query (SQ) algorithm to learn, and similarly for GD-like algorithms (Abbe, Kamath, et al., 2021); these bounds hold irrespective of the type of neural network architectures used.

A more architecture-dependent lower-bound is derived in (Abbe and Sandon, 2020b), where the junk-flow is essentially used as replacement of the number of queries, and which depends on the type of architecture and initialization albeit being implicit. In (Shalev-Shwartz and Malach, 2021), a separation between fully connected and CNN architectures is obtained, showing that certain target functions have a locality property and are better learned by the latter architecture. In a different setting, (Tan et al., 2021) gives a generalization lower bound for decision trees on additive generative models, proving that decision trees are statistically inefficient at estimating additive regression functions. However, none of the bounds in these works give an explicit figure of merit to measure the suitability of a neural network architecture for a target.

One can interpret such bounds, especially the one in (Abbe and Sandon, 2020b), as follows. If the function class is such that for two functions F, F' sampled randomly from the class, the typical correlation is not noticeable, i.e., if the cross-predictability (CP) is given by

$$\mathrm{CP}(F, F') := \mathbb{E}_{F,F'} \langle F, F' \rangle^2 = d^{-\omega_d(1)}, \tag{3.1}$$

(where we denoted by $\langle . \rangle$ the L^2-scalar product, namely, for some input distribution $P_{\mathscr{X}}$, $\langle f, g \rangle = \mathbb{E}_{x \sim P_{\mathscr{X}}}[f(x)g(x)]$ and by $\omega_d(1)$ any sequence that is diverging to ∞ as $d \to \infty$), then GD with polynomial precision and on a polynomial horizon will not be able to identify the target function with an inverse polynomial accuracy (weak learning), because at no time the algorithm will approach a good approximation of the target function; i.e. the gradients stay essentially agnostic to the target.

Instead, here we focus on a specific function — rather than a function class — and on a specific architecture and initialization. One can engineer a function class from a specific function if the initial architecture has some distribution symmetry. In such case, if the original function is learnable, then its orbit under the group of symmetry must also be learnable, and thus lower bounds based on the cross-predictability or statistical dimension of the orbit can be used. Such lower bounds are no longer applying to any architecture but exploit the symmetry of the architecture, however they still require knowledge of the target function in order to define the orbit.

Our goal is to depart from the setting where we know the target function and thus can analyze the orbit directly. Instead, we would like to have a ‘proxy’ that depends on the underlying target function and the initialized neural net $\mathrm{NN}(x;\theta^0)$ at hand, where the set of weights at time zero θ^0 are drawn according to some distribution. In (Abbe and Sandon, 2020a), the following proposal is made (the precise statement will appear below): can we replace the correlation among a function class by the correlation between a target function and an initialized net in order to have a necessary requirement for learning, i.e., if

$$\mathbb{E}_{\theta^0}\langle f, \mathrm{NN}(.;\theta^0)\rangle^2 = d^{-\omega_d(1)}, \tag{3.2}$$

or in other words, if at initialization the neural net correlates negligibly with the target function, is it still possible for GD to learn[I] the function f if the number of epochs of GD is polynomial? We next formalize the question further and provide an answer to it.

Note the difference between (3.1) and (3.2): in (3.1) it is the class of functions that is too poorly correlated for *any* SQ algorithm to efficiently learn; in (3.2) it is the specific network initialization that is too poorly correlated with the specific target in order for GD to efficiently learn.

While previous works and the proof presented in this Chapter rely on creating the orbit of a target function using the network symmetries and then arguing from the complexity of the orbit (using cross-predictability Abbe and Sandon, 2020a), we believe that the INAL approach can be fruitful in additional contexts. In fact, the orbit approaches have two drawbacks: (1) they cannot give lower-bounds on functions like the full parity[II] that have no complex orbit (in fact the orbit of the full parity is itself under permutation symmetries), (2) to estimate the complexity measure of the orbit class (e.g., the cross-predictability) from a sample set without full access to the target function, one needs labels of data points under the group action that defines the orbit (e.g., permutations), and these may not always be available from an arbitrary sample set. In contrast, (i) the INAL can still be small for the full parity function on certain symmetric neural networks, suggesting that in such cases the full parity is not learnable[III], (ii) the INAL can always be estimated from a random i.i.d. sample set, using basic Monte Carlo simulations (as used in our experiments, see Section 3.1.4).

While the notion of INAL makes sense for any input distribution, our theoretical results are proved in a more limited setting of Boolean functions with uniform inputs. This follows the approach that has been taken in (Abbe and Sandon, 2020b) and we made that choice for similar reasons. Furthermore, any computer-encoded function is eventually Boolean and major part of the PAC learning theory has indeed focused on Boolean functions (we refer to (Shalev-Shwartz and Ben-David, 2014) for more on this subject). We nonetheless expect that the footprints of the proofs derived in this Section will apply to inputs that are iid Gaussians or spherical, using different basis than the Fourier-Walsh one.

[I]Even with just an inverse polynomial accuracy, a.k.a., weak learning.

[II]we call full parity the function $f:\{\pm1\}^d \to \{\pm1\}$ s.t. $f(x)=\prod_{i=1}^d x_i$.

[III]This is not proven in this Section, due to a specific proof technique, but in Section 3.2.4 we prove that, under appropriate hypothesis, this result holds.

Our general strategy in obtaining such a result is as follows: we first show that for the type of architecture considered, a low initial alignment (INAL) implies that the implicit target function is essentially high-degree in its Fourier basis; this part is specific to the architecture and the low INAL property. We next use the symmetry of the initialization to conclude that learning under such high-degree Fourier requirement implies learning a low CP class, and thus conclude by leveraging the results from (Abbe and Sandon, 2020b). Finally, we do some experiments with the types of architecture used in our formal results, but also with convolutional neural nets to test the robustness of the original conjecture. We observe that generally the INAL gives a decent proxy for the difficulty to learn (lower INAL gives lower learning accuracy). While this goes beyond the scope of this Section — which is to obtain a first rigorous validation of the INAL conjecture for standard fully connected neural nets — we believe that the numerical simulations give some motivations to pursue the study of the INAL in a more general setting.

3.1.1 Definitions and Theoretical Contributions

For the purposes of our definition, a neural network NN consists of a set of neurons V_{NN}, a random variable $\theta^0 \in \mathbb{R}^k$ which corresponds to the initialization and a collection of functions $\mathrm{NN}^{(v)}(.;\theta^0) : \mathbb{R}^d \to \mathbb{R}$ indexed with $v \in V_{\mathrm{NN}}$, representing the outputs of neurons in the network. The Initial Alignment (INAL) is defined as the average squared correlation between the target function and any of the neurons at initialization:

Definition 3 (Initial Alignment (INAL))**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a function and $P_{\mathcal{X}}$ a distribution on $\mathbb{R}^d$. Let* NN *be a neural network with neuron set V_{NN} and random initialization θ^0. Then, the* INAL *is defined as*

$$\mathrm{INAL}(f,\mathrm{NN}) := \max_{v \in V_{\mathrm{NN}}} \mathbb{E}_{\theta^0} \langle f, \mathrm{NN}^{(v)}(.;\theta^0)\rangle^2, \tag{3.3}$$

where we denoted by $\langle . \rangle$ the L^2-scalar product, namely $\langle f, g\rangle = \mathbb{E}_{x\sim P_{\mathcal{X}}}[f(x)g(x)]$.

While the above definition makes sense for any neural network architecture, here we focus on fully connected networks. Thus, in the following NN will denote a fully connected neural network. Our main thesis is that in many settings a small INAL is bad news: If at initialization there is no noticeable correlation between any of the neurons and the target function, the GD-trained neural network will not be able to recover such correlation during training in polynomial time.

Of particular interest to us is the notion of INAL for a single neuron with activation σ and normalized Gaussian initialization.

Definition 4. *Let $f : \mathbb{R}^d \to \mathbb{R}$, $\sigma : \mathbb{R} \to \mathbb{R}$ and let $P_{\mathcal{X}}$ be a distribution on $\mathbb{R}^d$. Then, we abuse* *the notation and write*

$$\mathrm{INAL}(f,\sigma) := \mathbb{E}_{w^d,b^d}\left[\left(\mathbb{E}_{x\sim P_{\mathscr{X}}} f(x)\sigma((w^d)^T x + b^d)\right)^2\right], \tag{3.4}$$

where w^d is a vector of iid $\mathcal{N}(0,1/d)$ Gaussians and b^d is another independent $\mathcal{N}(0,1/d)$ Gaussian. In the following, for readability, we will write $w = w^d$ and $b = b^d$, omitting the dependence on the input dimension d.

In the following, we say that a function $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is *noticeable* if there exists $c \in \mathbb{N}$ such that $g(d) = \Omega(d^{-c})$. On the other hand, we say that g is *negligible* if $\lim_{d\to\infty} d^c g(d) = 0$ for every $c \in \mathbb{N}$ (which we also write $g(d) = d^{-\omega_d(1)}$).

Definition 5 (Weak learning). *Let $(f_d)_{d\in\mathbb{N}}$ be a sequence of functions such that $f_d : \mathbb{R}^d \to \mathbb{R}$ and (P_d) a sequence of probability distributions on $\mathbb{R}^d$. Let (A_d) be a family of randomized algorithms such that A_d outputs a function $\mathrm{NN}_d : \mathbb{R}^d \to \mathbb{R}$. Then, we say that A_d* weakly learns *f_d if the function*

$$g(d) := \left|\mathbb{E}_{x\sim\mathscr{P}_d,A_d}[f_d(x)\cdot \mathrm{NN}_d(x)]\right| \tag{3.5}$$

is noticeable.

In this Chapter, we follow the example of (Abbe and Sandon, 2020b) and focus on *Boolean* functions with inputs and outputs in $\{\pm1\}$. We consider sequences of Boolean functions $f_d : \{\pm1\}^d \to \{\pm1\}$, with the uniform input distribution $\mathscr{U}^d$, meaning that if $x \sim \mathscr{U}^d$, then for all $i \in [d]$, $x_i \overset{iid}{\sim} \mathrm{Rad}(1/2)$. We focus on fully connected neural networks with activation function σ, and trained by noisy GD — this means GD where the gradient's magnitude per precision noise is polynomially bounded, as commonly considered in statistical query algorithms (Blum et al., 1994; Kearns, 1998) and GD learning (Abbe, Kamath, et al., 2021; Abbe and Sandon, 2020b; Malach et al., 2021); see (1.9) or Remark 4 for a remainder the definition. We consider activation functions that satisfy the following conditions.

Definition 6 (Expressive activation). *We say that a function $\sigma : \mathbb{R} \to \mathbb{R}$ is* expressive *if it satisfies the following conditions:*

a) σ is measurable and polynomially bounded *i.e. there exists $C, c > 0$ such that $|\sigma(x)| \leq Cx^c + C$ for all $x \in \mathbb{R}$.*

b) Let the Gaussian smoothing of σ be defined as $\Sigma(t) := \mathbb{E}_{Y\sim\mathcal{N}(0,1)}[\sigma(Y+t)]$. For each $m \in \mathbb{N}$ either $\Sigma^{(m)}(0) \neq 0$ or $\Sigma^{(m+1)}(0) \neq 0$ (where $\Sigma^{(m)}$ denotes the m-th derivative of Σ).

Remark 1. *i) Note that we have the identities $d_m = \frac{\Sigma^{(m)}(0)}{m!}$, and $\sigma = \sum_{m=0}^{\infty} d_m H_m$, where H_m are the probabilist's Hermite polynomials (see Appendix C.3 for background on Hermite polynomials). Therefore, an equivalent statement of the second condition in*

Definition 6 is that there are no two or more consecutive zeros in the Hermite expansion of σ.

ii) *Many functions are expressive, including ReLU and sign (see Appendix B.2 for the proofs of those two cases).*

iii) *On the other hand, it turns out that polynomials are* not *expressive, as they do not satisfy point b). This is necessary for our hardness results to hold, since for an activation function P which is a polynomial of degree k and M a monomial of degree $k+1$ it can be checked that* $\mathrm{INAL}(M,P)=0$, *but constant-degree monomials are learnable by GD.*

Let us give one more definition before stating our main theorem.

Definition 7 (D-Extension)**.** *For a function $f:\mathbb{R}^d\to\mathbb{R}$ and for $D>d$, we define its* D-extension $\overline{f}:\mathbb{R}^D\to\mathbb{R}$ *as*

$$\overline{f}(x_1,x_2,...,x_d,x_{d+1},x_{d+2},...,x_D)=f(x_1,x_2,...,x_d). \tag{3.6}$$

In words, $\overline{f}$ is a D-dimensional function whose first d input coordinates correspond to the d input coordinates of f, and the other coordinates are dummy variables. We can now state our main result which connects INAL and weak learning.

Theorem 3 (Main theorem, informal)**.** *Let σ be an expressive activation function and (f_d) a sequence of Boolean functions with uniform distribution on $\{\pm1\}^d$. If* $\mathrm{INAL}(f_d,\sigma)$ *is negligible, then, for every $\epsilon>0$, the $d^{1+\epsilon}$-extension of f_d is* not *weakly learnable by any* $\mathrm{poly}(d)$*-sized fully-connected neural networks with i.i.d. initialization and* $\mathrm{poly}(d)$*-number of steps of noisy gradient descent.*

Remark 2. *Theorem 3 says that Boolean functions that have negligible correlation for* some *expressive activation and Gaussian i.i.d. initialization, cannot be learned by neural networks utilizing* any *activation on a fully-connected architecture and any i.i.d. initialization.*

Remark 3. *Consider a sequence of neural networks* (NN_d) *utilizing an expressive activation σ. We believe that the notion of* $\mathrm{INAL}(f_d,\mathrm{NN}_d)$ *is relevant to characterizing if a family of Boolean functions (f_d) is weakly learnable by noisy GD on those neural networks. On the one hand, if* $\mathrm{INAL}(f_d,\mathrm{NN}_d)$ *is noticeable, then at initialization there exists a neuron from which a weak correlation with f_d can be extracted. Therefore, in a sense weak learning is achieved at initialization.*

On the other hand, assume additionally that the architecture is such that there exists a neuron computing $\sigma(w^Tx+b)$, where x is the input and (w,b) are initialized as i.i.d. $\mathcal{N}(0,1/d)$ Gaussians. (In other words, there exists a fully-connected neuron in the first hidden layer.) Then, by definition of INAL, if $\mathrm{INAL}(f_d,\mathrm{NN}_d)$ *is negligible, then also* $\mathrm{INAL}(f_d,\sigma)$ *is negligible. Accordingly, by Theorem 3, an extension of (f_d) is not weakly learnable.*

While we do not have a proof, we suspect that a similar property might hold also for some other architectures and initializations.

Note that we obtain hardness only for an extension of f_d, rather than for the original function. Interestingly, in some settings GD can learn the function, while the $2d$-extension of the same function is hard to learn[IV]. However, in Section 3.2 we show that such examples cannot be constructed for fully connected networks with Gaussian initialization, trained with the hinge loss. Thus, in the cases covered by Section 3.2 the input extension can be removed.

3.1.2 Formal Results

In this section, we write precise statements of our theorems. For this, we need a couple of more definitions.

Definition 8 (Cross-Predictability, Abbe and Sandon, 2020a)**.** *Let $P_{\mathcal{F}}$ be a distribution over functions from $\mathbb{R}^d$ to $\mathbb{R}$ and $P_{\mathcal{X}}$ a distribution over $\mathbb{R}^d$. Then,*

$$\mathrm{CP}(P_{\mathcal{F}}, P_{\mathcal{X}}) = \mathbb{E}_{F,F'\sim P_{\mathcal{F}}}[\mathbb{E}_{X\sim P_{\mathcal{X}}}[F(X)F'(X)]^2]\,. \tag{3.7}$$

Definition 9 (Orbit)**.** *For $f : \mathbb{R}^d \to \mathbb{R}$ and a permutation $\pi \in S_d$, we let $(f\circ\pi)(x) = f(x_{\pi(1)}, \dots, x_{\pi(d)})$. Then, we define the* orbit *of f as*

$$\mathrm{orb}(f) := \{f \circ \pi : \pi \in S_d\}\,. \tag{3.8}$$

Let us now give the full statement of our main theorem.

Theorem 4. *Let (f_d) be a sequence of Boolean functions with $f_d : \{\pm1\}^d \to \{\pm1\}$ and $x \sim \mathcal{U}^d$ and let σ be an expressive activation. If $\mathrm{INAL}(f_d,\sigma)$ is negligible, then, for every $\epsilon > 0$, the cross predictability $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D)$ is negligible, where $D = d^{1+\epsilon}$ and $\mathrm{orb}(\overline{f_d})$ denotes (uniform distribution on) the orbit of the D-extension of f_d.*

More precisely, if $\mathrm{INAL}(f_d,\sigma) = O(d^{-c})$, then $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D) = O(d^{-\frac{\epsilon}{1+\epsilon}(c-1)})$.

Applying (Abbe and Sandon, 2020b,[Theorem 3]) to Theorem 4 implies the following corollary. We refer to Appendix B.7 for additional clarifications on the notion of a fully connected neural net.

Corollary 3. *Let f_d and σ be as in Theorem 4 with negligible $\mathrm{INAL}(f_d,\sigma)$ and let $\epsilon > 0$ with $D = d^{1+\epsilon}$ and $\overline{f_d}$ denote the D-extension of f_d. Let $\mathrm{NN} = (\mathrm{NN}_d)$ be any sequence of fully connected neural nets of polynomial size. Then, for any iid initializaton, and any polynomial bounds on the learning rate, learning time $T = (T_d)$, noise level and overflow range, the noisy*

[IV]For example, for the Boolean parity function $\chi_{[d]}(x) = \prod_{i=1}^d x_i$ with both the input distribution and the weight initialization iid uniform in $\{\pm1\}$ (Abbe and Boix-Adserà, 2022).

GD algorithm after T steps of training outputs a neural net $\mathrm{NN}(.;\theta^T)$ *such that the correlation*

$$g(d) := \left|\mathbb{E}_{\mathrm{NN}(.;\theta^T)}\langle \mathrm{NN}(.;\theta^T), \overline{f_d}\rangle\right| \tag{3.9}$$

is negligible.

More precisely, if $\mathrm{INAL}(f_d,\sigma) = O(d^{-c})$*, then for a noisy GD run for T steps on a fully connected neural network with P edges, with learning rate γ, overflow range A and noise level τ it holds that*

$$g(d) = O\left(\frac{\gamma T\sqrt{P}A}{\tau}\cdot d^{-\frac{\epsilon}{4(1+\epsilon)}(c-1)}\right). \tag{3.10}$$

Remark 4. *In the result above, the neural net can have any feed-forward architecture with layers of fully-connected neurons and any activation such that the gradients are almost surely well-defined. The initialization can be i.i.d. from any distribution (which can depend on d). We remark that the result of Corollary 3 can be strengthen to apply to any initialization such that the distribution of the weights in the first layer is invariant under permutations of input neurons. We refer to Appendix B.7 for more details.*

The algorithm considered is noisy gradient descent[V] *using any differentiable loss function, meaning that at every step an i.i.d. $\mathcal{N}(0,\tau^2)$ noise vector is added to all components of the gradient, where τ is called the* noise level. *Furthermore, every component of the gradient during the execution of the algorithm whose evaluation exceeds the* gradient range *A in absolute value is clipped to A or $-A$, respectively. This covers in particular the bounded 'precision model' of (Abbe, Kamath, et al., 2021).*

For the purposes of function $g(d)$, it is assumed that the neural network outputs a guess in $\{\pm1\}$ using any form of thresholding (e.g., the sign function) on the value of the output neuron. See (Abbe and Sandon, 2020b[Section 2.3.1]).

3.1.3 Proof of Main Theorem

In this section we sketch the proof of Theorem 4. We first state basic definitions from Boolean function analysis, then we give a short outline of the proof, and then we state main propositions used in the proof. Finally, we show how the propositions are combined to prove Theorem 4 and Corollary 3. Further proofs and details are in the appendices.

We recall some notions of Boolean analysis, mainly taken from Chapters 1,2 of (O'Donnell, 2014). For every $f : \{\pm1\}^n \to \mathbb{R}$ we denote its Fourier expansion as

$$f(x) = \sum_{S\subseteq[n]} \hat{f}(S)\chi_S(x), \tag{3.11}$$

[V] In fact, it can be SGD with batch size B for large enough B.

where $\chi_S(x) = \prod_{i \in S} x_i$ are the standard Fourier basis elements and $\hat{f}(S)$ are the Fourier coefficients of f, defined as $\hat{f}(S) = \langle f, \chi_S \rangle$. We denote by

$$W^k(f) = \sum_{S:|S|=k} \hat{f}(S)^2 \tag{3.12}$$

$$W^{\leq k}(f) = \sum_{S:|S|\leq k} \hat{f}(S)^2 , \tag{3.13}$$

the total weight of the Fourier coefficients of f at degree k (respectively up to degree k).

Definition 10 (High-Degree)**.** *We say that a family of functions $f_d : \{\pm 1\}^d \to \mathbb{R}$ is "high-degree" if for any fixed k, $W^{\leq k}(f_d)$ is negligible.*

Proof Outline of Theorem 4.

1. We initially restrict our attention to the basis Fourier elements, i.e. the monomials $\chi_S(x) := \prod_{i \in S} x_i$ for $S \in [d]$. We consider the single-neuron alignments INAL(χ_S, σ) for expressive activations. We prove that these INALs are noticeable for constant degree monomials (Proposition 3).

2. For a general $f : \{\pm 1\}^d \to \mathbb{R}$ we show that the initial alignment between f and a single-neuron architecture can be computed from its Fourier expansion (Proposition 4). As a consequence, for any expressive σ, if INAL(f, σ) is negligible, then f is high-degree (Corollary 4).

3. We construct the extension of f and take its orbit orb($\overline{f}$). Since the extension has a sparse structure of its Fourier coefficients, that guarantees that the cross-predictability of orb($\overline{f}$) is negligible (Proposition 5).

4. In order to prove Corollary 3, we invoke the lower bound of Abbe and Sandon, 2020b (Theorem 5) applied to the class orb($\overline{f}$) .

A crucial property of the expressive activations is that they correlate with constant-degree monomials. To emphasize this, we introduce another definition.

Definition 11. *An activation σ is* correlating *if for every k, the sequence* INAL(χ_k, σ) *is noticeable, where we think of $\chi_k(x) = \prod_{i=1}^k x_i$ as a sequence of Boolean functions for every input dimension $d \geq k$.*

Furthermore, if there exists c such that for every k it holds INAL(χ_k, σ) $= \Omega(d^{-(k+c)})$*, then we say that σ is c*-strongly correlating.

Proposition 3. *If σ is expressive (according to Definition 6), then it is 1-strongly correlating.*

The proof of Proposition 3 is our main technical contribution. Since the magnitude of the correlations is quite small (in general, of the order d^{-k} for monomials of degree k), careful calculations are required to establish our lower bounds.

In fact, we conjecture that any polynomially bounded function that is not a polynomial (almost everywhere) is correlating.

Then, we show that $\mathrm{INAL}(f,\sigma)$ decomposes into monomial INALs according to its Fourier coefficients:

Proposition 4. *For any $f:\{\pm 1\}^d \to \mathbb{R}$ and any activation σ,*

$$\mathrm{INAL}(f,\sigma) := \sum_{T\in[d]} \hat{f}(T)^2\, \mathrm{INAL}(\chi_T,\sigma)\,. \tag{3.14}$$

As a corollary, functions with negligible INAL on correlating activations are high-degree:

Corollary 4. *Let σ be an activation with $\mathrm{INAL}(\chi_{k'},\sigma) = \Omega(d^{-k_0})$ for $k' = 0,1,\dots,k$. Then, $W^{\le k}(f_d) \le \mathrm{INAL}(f_d,\sigma) O(d^{k_0})$.*

In particular, if σ is correlating and $\mathrm{INAL}(f_d,\sigma)$ is negligible, then (f_d) is high degree.

Finally, the cross-predictability of $\mathrm{orb}(\overline{f_d})$ is negligible for high degree functions.

Proposition 5. *Let $\epsilon > 0$ and (f_d) a family of Boolean functions. Let $(\overline{f_d})$ denote the family of D-extensions of f_d for $D = d^{1+\epsilon}$, and consider the uniform distribution on its orbit.*

If (f_d) is high degree, then $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D)$ is negligible. Furthermore, if for some universal c and every fixed k it holds $W^{\le k}(f_d) = O(d^{k-c})$, then $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D) = O(d^{-\frac{\epsilon}{1+\epsilon}\cdot c})$.

Theorem 5 (Abbe and Sandon, 2020b, informal)**.** *If the cross-predictability of a class of functions is negligible, then noisy GD cannot learn it in poly-time.*

We refer to Appendix B for the proofs of the results presented in this Section.

3.1.4 Experiments

In this section we present a few experiments to show how the INAL can be estimated in practice. Our theoretical results connect the performance of GD to the Fourier spectrum of the target function. However, in applications we are usually given a dataset with data points and labels, rather than an explicit target function, and it may not be trivial to infer the Fourier properties of the function associated to the data. Conveniently, the INAL can be estimated with sufficient datapoints and labels, and do not need an explicit target.

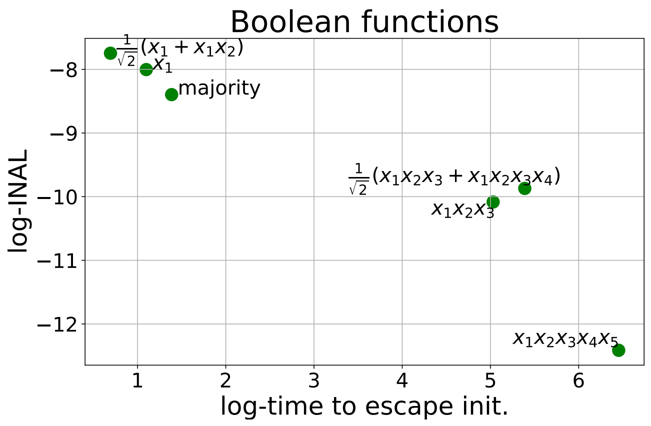

Figure 3.1: Comparison between INAL and time needed to "escape the initialization" (e.g. achieve non-trivial learning) for several Boolean tasks. We estimate the INAL between each target function and a 2-layers ReLU fully connected neural network with input size input size $d = 1000$ and normalized gaussian initialization. We then the network with SGD batch size 64.

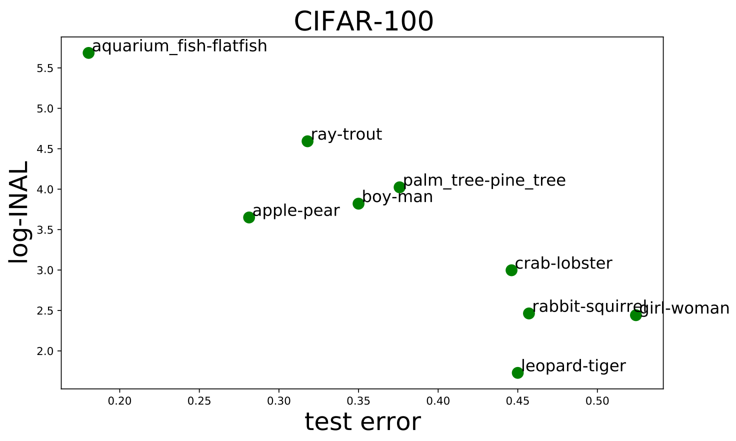

Figure 3.2: Comparison between INAL and time needed to "escape the initialization" (e.g. achieve non-trivial learning) for binary classification tasks in the CIFAR100 dataset. We estimate the INAL between each target tasks and a shallow CNN. We then the network with SGD batch size 64.

Experiments on Boolean functions. In our first experiment, we consider several Boolean functions, such as the majority-vote over the whole input space ($\mathrm{Maj}_d(x) := \mathrm{sgn}(\sum_{i=1}^{d} x_i)$), monomials of different degree and sums of monomials, on an input space of dimension 1000. We take a 2-layer fully connected neural network with ReLU activations and normalised Gaussian iid initialization (according to the setting of our theoretical results), and we train it with SGD with batch-size 64, until the network achieves non-trivial correlation (i.e. accuracy 1/10) with the target function. We call the number of time-steps needed to achieve such non-trivial correlation the *time to escape the initialization.* On the other hand, we estimate the INAL between each of the targets and the neural network, through Monte-Carlo. Our observations confirm our theoretical claim, i.e. that INAL is a good proxy of the time needed to achieve weak learning (Figure 3.1).

Experiments on CIFAR. Given a dataset $D = (x_s, y_s)_{s\in[m]}$, where $x_s \in \mathbb{R}^d$, and $y_s \in \mathbb{R}$, and given a randomly initialized neural network $\mathrm{NN}(.;\theta^0)$ with θ^0 drawn from some distribution, we can estimate the initial alignment between the network and the target function associated to the dataset as

$$\max_{v\in V_{NN}} \mathbb{E}_{\theta^0}\left[\left(\frac{1}{m}\sum_{s=1}^{m} y_s \cdot \mathrm{NN}^{(v)}(x_s;\theta^0)\right)^2\right], \tag{3.15}$$

where the outer expectation can be performed through Monte-Carlo approximation.

We ran experiments on the CIFAR100 dataset. We split the dataset into different pairs of classes, corresponding to different binary classification tasks. We choose pairs of classes that are intuitively hard to distinguish. We take a CNN with 1 VGG block and ReLU activation, and for each task, we train the network with SGD with batch-size 64, and we estimate the INAL according to (3.15). We notice that also in this setting (not covered by our theoretical results), the INAL and the generalization accuracy present some correlation, and a significant difference in the INAL corresponds to a significant difference in the accuracy achieved after training. This may give some motivation to study the INAL beyond the fully connected setting.

3.2 Initial Gradient Alignment

The result of Section 3.1 presents few limitations. Firstly, Theorem 4 is only applicable to uniform Boolean inputs. Secondly, the bound is independent of the network's initialization scale and architecture, and depends solely on the complexity of the target task's orbit and the network's number of parameters. Finally, it provides hardness of learning only for an extension of the target function f, preventing any application to functions whose orbits are SQ learnable.

In this Section, we address these limitations, by narrowing our focus to fully connected ReLU networks of finite depth trained with the noisy-GD algorithm (defined in (1.9)) on the *correlation loss*[VI].

For ease of notation, we denote by

$$G_f(\theta) := \mathbb{E}_{x\sim\mathscr{D}}\big[f(x)\cdot\nabla_\theta \operatorname{NN}(x;\theta)\big] \tag{3.16}$$

the alignment between f and the gradients of the neural network at θ, where $\mathscr{D}$ is an arbitrary input distribution. We focus on functions $f:\mathbb{R}^d\to\{\pm1\}$ (binary classification tasks). We prove the following results:

1. If:
$$\mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^2 = d^{-\omega(1)}, \tag{3.17}$$
then noisy-GD on a fully connected ReLU[VII] network of finite depth with Gaussian i.i.d. initialization cannot efficiently learn f (Theorem 6).
2. If:
$$K_\mu\cdot\mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^4 = d^{-\omega(1)}, \tag{3.18}$$
where K_μ is given in (3.25), then noisy-GD on any network with parameters initialized from a distribution μ that satisfies Assumption 2, cannot efficiently learn f (Theorem 7).
3. Noisy-GD on a 2-layer ReLU network with Gaussian i.i.d. initialization, cannot learn high-degree Boolean functions (as defined in Def. 10) with uniform inputs (Corollary 5).

Let us make few remarks, before defining our results formally.

Remark 5. *Similar quantities to the ones in* (3.17)-(3.18) *are proposed in (Mok et al., 2022; Ortiz-Jiménez et al., 2021) as measures to predict the generalization performance of neural networks after training. In particular, one can rewrite*

$$\|G_f(\theta^0)\|_2^2 = \mathbb{E}_{x,x'\overset{\text{iid}}{\sim}\mathscr{D}}\big[f(x)f(x')\Theta_{\theta^0}(x,x')\big], \tag{3.19}$$

where $\Theta_{\theta^0}(x,x')$ denotes the Neural Tangent Kernel (NTK) (see Jacot et al., 2018) at initialization, and observe that the left-hand-side of (3.17) *is the expectation over θ^0 of the Label Gradient Alignment (LGA) introduced in (Mok et al., 2022). In this Section, we thus give some theoretical insights related to the empirical observations of the aforementioned papers.*

[VI]Our results hold for noisy-GD on the *hinge-loss*, with an additional assumption on the initialization.
[VII]Our proof works for all activations that satisfy a homogeneity assumption, see Def. 12.

Remark 6. *Point 3 gives a separation between SQ algorithms and noisy-GD on the specific setting considered, in the following sense. Let us assume that the target f is the parity function on all bits $f(x) = \prod_{i=1}^d x_i$. For SQ algorithms, the class of parities of degree d is trivially learnable, as it contains only f. We prove that in the setting considered (2-layer* ReLU *nets with Gaussian init. and correlation loss) this is not the case, and f cannot be learned in polynomial time. It has been proven (Abbe and Boix-Adserà, 2022) that f can be learned in $O(1)$ steps by a 2-layer* ReLU *network with weights initialized i.i.d* $\mathrm{Rad}(1/2)$. *The characterization of the set of initializations that allow (and do not allow) GD to learn the full parity and the understanding of whether the Rademacher initialization is 'robust', are left to future work.*

3.2.1 Definitions and Formal Results

I.i.d. Gaussian Initialization. For our first Theorem, we focus on fully-connected networks of finite depth. For a network of depth L, we denote by P the total number of parameters in the network and by $\theta \in \mathbb{R}^P$ the vector containing all parameters of the network. We use the following notation:

$$x^{(1)}(\theta) = W^{(1)}x + b^{(1)} \tag{3.20}$$

$$x^{(l)}(\theta) = W^{(l)}\sigma(x^{(l-1)}(\theta)), \qquad l = 2, \dots, L, \tag{3.21}$$

For simplicity, we consider fully connected networks with one bias vector in the first layer, but we believe that, with a more involved argument, one could extend the proof and include bias vectors in all layers. We denote the network function as $\mathrm{NN}(x;\theta) = x^{(L)}(\theta)$. We assume that each parameter of the network is independently initialized as

$$\theta_p^0 \sim \mathcal{N}(0, v_{l_p}^2), \tag{3.22}$$

where l_p denotes the layer of parameter θ_p, for $p \in [P]$.

We assume that the activation σ satisfies the following homogeneity property.

Definition 12 (H-Strongly Homogeneous.)**.** *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be an activation function. We say that σ is H-strongly homogeneous iff for all $x \in \mathbb{R}$:*

$$\sigma(Cx) = C^H \sigma(x). \tag{3.23}$$

Example 3. $\mathrm{ReLU}(x) = \max\{0, x\}$ *is 1-strongly homogeneous. x^k is k-strongly homogeneous, for all $k \in \mathbb{N}$.*

Furthermore, we need the following assumption on the neural network output at initialization. We are now ready to state our main Theorem on networks with i.i.d. Gaussian initialization.

Theorem 6 (Gaussian init.)**.** *Let* $\mathrm{NN}(x;\theta)$ *be a fully connected network of depth L, with*

H-strongly homogeneous activation and with weights initialized according to (3.22). *Let $f : \mathbb{R}^d \to \{\pm 1\}$ be a target function. Let* $\mathrm{NN}(x;\theta^T)$ *be the output of the noisy-GD algorithm after T steps of training with the correlation loss. Then,*

$$\mathbb{P}(\mathrm{NN}(x;\theta^T) = f(x)) \leq \frac{1}{2} + \frac{T}{2\tau}\prod_{l=1}^{L}\left(1+\frac{T\gamma^2\tau^2}{v_l^2}\right)^H \cdot \mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^2. \tag{3.24}$$

For the purposes of Theorem 6, we assume that $\mathrm{NN}(x;\theta)$ is the network's guess in $\{\pm 1\}$, obtained using any form of thresholding (e.g. the sgn function) on the output neuron. We remark that the result extend to noisy-GD with the *hinge loss*, instead of the correlation loss, if the initialization satisfy the following assumption with $K = \prod_{l=1}^{L}\left(1+\frac{T\gamma^2\tau^2}{v_l^2}\right)^H$.

Assumption 1 (K-bounded initialization)**.** *We say that a network is initialized with K-bounded initialization if for all x, with probability at least* $1-\exp(-d\delta)$ *over θ^0, for some $\delta > 0$, $|\mathrm{NN}(x;\theta^0)| < K$.*

Beyond i.i.d Gaussian Initialization. We prove a second result that holds for a different set of initial distributions, and for general architectures (beyond fully connected). Let $\mathrm{NN}(x;\theta)$ be a neural network of any architecture, with P parameters initialized according to a distribution μ supported in $\mathbb{R}^P$. We assume that μ satisfies the following assumption.

Assumption 2. *Let μ be a probability measure on $\mathbb{R}^P$. We assume that μ is such that*

$$K_\mu := \int_{\mathbb{R}^P}\frac{1}{\phi_\mu(\xi)}\left(\int_{\mathbb{R}^P}\phi_{\mathcal{N}}(\xi-\psi)\phi_\mu(\psi)d\psi\right)^2 d\xi < \infty, \tag{3.25}$$

where ϕ_μ denotes the density function of μ and $\phi_{\mathcal{N}}$ denotes the density of a $\mathcal{N}(0,\gamma^2 T\tau^2\mathbb{I})$ *random vector.*

Theorem 7 (μ init.)**.** *Let μ be a distribution in $\mathbb{R}^P$ that satisfies Assumption 2. Let* $\mathrm{NN}(x;\theta)$ *be a neural network of any architecture with weights initialized according to μ. Let $f : \mathbb{R}^d \to \{\pm 1\}$ be a target function. Let* $\mathrm{NN}(x;\theta^T)$ *be the output of the noisy-GD algorithm after T steps of training with the correlation loss. Then,*

$$\mathbb{P}(\mathrm{NN}(x;\theta^T) = f(x)) \leq \frac{1}{2} + \frac{T}{2\tau}K_\mu^{1/2}\cdot\left(\mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^4\right)^{1/2}, \tag{3.26}$$

where K_μ is given by Assumption 2.

Theorem 7 extends to noisy-GD with the hinge loss on networks with 1-bounded initialization.

Remark 7 (Gaussian Initialization)**.** *In Appendix C.1.5 we show that if θ^0 is initialized according to* (3.22), *Assumption 2 holds if and only if $v_l^2 \geq \gamma^2 t\tau^2$ for all $l \in [L]$ and in such* *case $K_\mu^2 = \prod_{l=1}^{L} \frac{v_l^2}{\sqrt{v_l^4 - \gamma^4 t^2 \tau^4}}$. Thus, the bound in Theorem 7 depends exponentially on the number of parameters (as opposed to the one in Theorem 6 that depends exponentially on the networks' depth) and on the 4-th moment of $G_f(\theta^0)$ (as opposed to the 2-th moment obtained in Theorem 6). On the other hand, Theorem 7 holds for all networks architectures and activations, while Theorem 6 is specific to fully connected and strongly homogeneous activations. Furthermore, Theorem 6 holds for non-iid Gaussian initialization.*

Remark 8 (Uniform Initialization)**.** *In Appendix C.1.5 we further show that the proof can be extended to cover some distributions μ with compact support, including in particular the* $\mathrm{Unif}[-1,1]^{\otimes P}$ *distribution.*

3.2.2 Proof Outline

For brevity, we denote the population gradient at θ for a target function f by

$$\Gamma_f(\theta) := \mathbb{E}_x\left[\nabla_\theta L(f,\theta,x)\right]. \tag{3.27}$$

To prove our results we couple the dynamics of the network's weights θ^t with the dynamics of the 'Junk-Flow'. The junk-flow is the dynamics of the parameters of a network trained on random labels.

Definition 13 (Junk-Flow)**.** *Let us define the junk-flow as the sequence $\psi^t \in \mathbb{R}^P$ that satisfies the following iterations:*

$$\psi^0 = \theta^0, \tag{3.28}$$

$$\psi^{t+1} = \psi^t - \gamma\left(\Gamma_r(\psi^t) + \xi^t\right), \tag{3.29}$$

where $\xi^t \overset{iid}{\sim} \mathcal{N}(0, \mathbb{1}\tau^2)$, $\Gamma_r(\psi^t)$ is the population gradient for the target function $r(x) =$ $\mathrm{Rad}(1/2)$ for all x (i.e. random labels) and γ, τ are respectively the learning rate and the noise-level of the noisy-GD algorithm used to train the network $\mathrm{NN}(x;\theta)$.

We couple θ^T and ψ^T in terms of the total variation distance. Let us look at the total variation distance between the law of θ^T and ψ^T, which, by abuse of notation, we denote by $\mathrm{TV}(\theta^T;\psi^T)$.

Lemma 1. *Let* $\mathrm{TV}(\theta^T;\psi^T)$ *be the total variation distance between the law of θ^T and ψ^T. Then,*

$$\mathrm{TV}(\theta^T;\psi^T) \le \frac{1}{2\tau}\sum_{t=0}^{T-1} \sqrt{\mathbb{E}_{\psi^t} \|\Gamma_f(\psi^t) - \Gamma_r(\psi^t)\|_2^2}. \tag{3.30}$$

The proof of Lemma 1 can be found in Appendix C.1. Our goal is to bound the right-hand-side of (3.30) in terms of moments of $\|G_f(\theta^0)\|_2$. We show these bounds in Section 3.2.3.

Recalling that $f : \mathbb{R}^P \to \{\pm 1\}$, we have

$$\mathbb{P}(\mathrm{NN}(x;\theta^T) = f(x)) \le \mathbb{P}(\mathrm{NN}(x;\psi^T) = f(x)) + \mathbb{E}_{\theta^0}\left[\mathrm{TV}(\theta^T;\psi^T)\right] \tag{3.31}$$

$$= \frac{1}{2} + \mathbb{E}_{\theta^0}\left[\mathrm{TV}(\theta^T;\psi^T)\right], \tag{3.32}$$

where in (3.31), we denote by $\mathrm{NN}(x;\theta^T)$ the guess in $\{\pm 1\}$ of the network.

3.2.3 Alignment along the Junk-Flow

We are now left with bounding the right-hand-side of (3.30) in terms of the moments of $\|G_f(\theta^0)\|_2$. In other words, we want to show that the junk-flow dynamics does not pick correlation with f along its trajectory.

I.i.d. Gaussian Initialization. Let θ^0 be initialized according to (3.22). Then,

$$\psi_p^t \sim \mathcal{N}(0, v_{l_p}^2 + t\gamma^2\tau^2), \tag{3.33}$$

where l_p denotes the layer of parameter p. Moreover, for all t, ψ_p^t is independent of ψ_q^t, for all $p \neq q$.

Proposition 6 (Gaussian init.)**.** *Let* $\mathrm{NN}(x;\theta)$ *be a network that satisfies the assumptions of Theorem 6. Then, for all* $t \in [T-1]$

$$\mathbb{E}_{\psi^t}\|\Gamma_f(\psi^t) - \Gamma_r(\psi^t)\|_2^2 \le \prod_{l=1}^{L}\left(1 + \frac{t\gamma^2\tau^2}{v_l^2}\right)^H \cdot \mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^2. \tag{3.34}$$

The proof of Proposition 6 is deferred to Appendix C.1.2.

Beyond i.i.d. Gaussian Initialization. Let us now assume that $\theta^0 \sim \mu$, for some probability measure μ supported on $\mathbb{R}^P$. We assume that the support of μ is the whole space $\mathbb{R}^P$. In this case, the parameters of the junk-flow at time t are given by,

$$\psi^t = \theta^0 + \mathcal{N}(0, \gamma^2 t\tau^2\mathbb{I}) \sim \mu * \mathcal{N}(0, \gamma^2 t\tau^2\mathbb{I}), \tag{3.35}$$

where by $\mu * \nu$ we denoted the convolution in $\mathbb{R}^P$, i.e. the probability measure that has density $\phi_{\mu*\nu}$ given by

$$\phi_{\mu*\nu}(\theta) = \int_{\mathbb{R}^P} \phi_\mu(\psi)\phi_\nu(\theta-\psi)d\psi, \tag{3.36}$$

where ϕ_μ, ϕ_ν are the densities of μ and ν respectively. In this case, we prove the following bound.

Proposition 7. *Let* $\mathrm{NN}(x;\theta)$ *be a network that satisfies the assumptions of Theorem 7. Then,*

$$\mathbb{E}_{\psi^t}\|\Gamma_f(\psi^t)-\Gamma_r(\psi^t)\|_2^2 \le K_\mu^{1/2}\cdot \mathbb{E}_{\theta^0}[\|G_f(\theta^0)\|_2^4]^{1/2}, \tag{3.37}$$

where K_μ *is given by* (3.25)*.*

The proof of Proposition 7 is deferred to Appendix C.1.4.

3.2.4 Application to High-Degree Boolean Functions

Let us focus on the specific case of Boolean target functions with inputs sampled uniformly from the hypercube. Let $\mathrm{NN}(x;\theta)$ be a 2-layers fully connected neural network with Gaussian initialization, according to (3.22). We assume the following condition on the activation function.

Assumption 3. *Let* $\sigma:\mathbb{R}\to\mathbb{R}$ *be an activation function in* $L^2(\mathcal{N}(0,1))$ *and let* $\hat{\sigma}(k)$*,* $k\in\mathbb{N}$*, denote the coefficients of its expansion in Hermite series (see C.3 for definition of Hermite coefficients). Let* σ' *be a sub-derivative of* σ*, and let* $\hat{\sigma}'(k)$ *be its Hermite coefficients. We assume that:*

$$\forall k:\hat{\sigma}(k)\neq 0, \qquad \hat{\sigma}(k)^2k^2 \ge \hat{\sigma}(k')^2k'^2 \qquad \forall k'\ge k, \tag{3.38}$$

$$\forall k:\hat{\sigma}'(k)\neq 0, \qquad \hat{\sigma}'(k)^2k^2 \ge \hat{\sigma}'(k')^2k'^2 \qquad \forall k'\ge k. \tag{3.39}$$

Example 4. $\mathrm{ReLU}(x)$ *satisfies Assumption 3 (see Lemmas 16 and 17) in Appendix.*

The following Proposition gives an upper bound on the second moment of $G_f(\theta^0)$, depending on the Fourier spectrum of f.

Proposition 8. *Let* $f:\{\pm1\}^d\to\{\pm1\}$*. Let* $W^{\le k}[f]$ *be the Fourier weight of* f *up to degree* k *(as defined in 3.13). Let* $\mathrm{NN}(x;\theta)$ *be a* 2*-layer neural network with activation that satisfies Assumption 3 and that is* H*-strongly homogeneous. Then, for all* $k\in\mathbb{N}$*,*

$$\mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^2 \le W^{\le k}[f]\cdot C + PV^{H-1}\frac{\pi^2}{6}\hat{\sigma}'(m')m'^2 + NV^H\frac{\pi^2}{6}\hat{\sigma}(m)m^2, \tag{3.40}$$

where $m:=\min\{k'\ge k:\hat{\sigma}(k')\neq 0\}$*,* $m':=\min\{k'\ge k:\hat{\sigma}'(k')\neq 0\}$*,* $V=v_1\sqrt{d+1}$*,* v_1 *is the variance of the initialization of the first layer's weights,* N *is the number of hidden neurons.*

The proof of Proposition 8 can be found in Appendix C.2.

Theorem 6 and Proposition 8 gives the following Corollary, which states that high-degree Boolean functions are not efficiently learnable by a 2-layer network with Gaussian initialization.

Corollary 5. *Let* $\mathrm{NN}(x;\theta)$ *be a* 2*-layer neural network with activation that satisfies Assumption 3 and that is* H*-strongly homogeneous. Let* θ^0 *be initialized from a Gaussian distribution (according to* (3.22)*). Let* f *be a Boolean high-degree function (as defined in Def. 10), and let the input distribution be* $\mathscr{D} = \mathrm{Unif}\{\pm 1\}^d$*. Then, the noisy-GD algorithm with the correlation loss after* T *steps of training outputs a network such that*

$$\mathbb{P}\big(\mathrm{NN}(x;\theta^T) = f(x)\big) \le \frac{1}{2} + d^{-\omega(1)}. \tag{3.41}$$

3.3 Noise Stability

Similar techniques to the ones presented in the previous Sections allow to prove a lower bound on the generalization error achieved by GD, or SGD with large batch-size, to a measure of the complexity of the task, namely the *Noise Stability* ($\mathrm{Stab}_\delta[f]$). This provides theoretical support to a conjecture made in (Zhang et al., 2021), based on empirical observations. The latter work used the noise sensitivity as a dual measure of the target function complexity, whereas we use here the noise stability, but the two measures can be easily related, as we explain below. Let us formally define the Noise Stability.

Definition 14 (Noise stability)**.** *Let* $f : \{\pm 1\}^d \to \mathbb{R}$ *and* $\delta \in [0,1/2]$*. Let* x *be uniformly distributed on the Boolean* d*-dimensional hypercube, and let* y *be formed from* x *by flipping each bit independently with probability* δ*. We define the* δ*-noise stability of* f *by*

$$\mathrm{Stab}_\delta[f] := \mathbb{E}_{(x,y)}[f(x)\cdot f(y)]. \tag{3.42}$$

Intuitively, $\mathrm{Stab}_\delta[f]$ measures how stable the output of f is to a perturbation that flips each input bit with probability δ, independently. The noise stability can be easily related to the noise sensitivity $\mathrm{NS}_\delta[f]$[VIII], used in (Zhang et al., 2021).

Example 5.
- *(Parities): For* $\chi_S(x) := \prod_{i\in S} x_i$*, with* $|S| = k$*, we have:*

$$\mathrm{Stab}_\delta(\chi_S) = (1-2\delta)^k. \tag{3.43}$$

- *(Majority): For* $\mathrm{Maj}_d(x) := \mathrm{sgn}\Big(\sum_{i=1}^d x_i\Big)$*, we have for* $d \to \infty$*:*

$$\mathrm{Stab}_\delta(\mathrm{Maj}_d) \sim \frac{2}{\pi}\arcsin(1-2\delta) \tag{3.44}$$

For parities, the noise stability decreases exponentially fast with the degree of the parity, thus high-degree parities are highly noise unstable functions. On the other hand, majority is a fairly noise stable function, as for all $\delta < 1/2$ the noise stability converges to a positive constant for large d.

[VIII]Specifically, for binary-valued functions, $\mathrm{NS}_\delta[f] = \frac{1}{2} - \frac{1}{2}\,\mathrm{Stab}_\delta[f]$.

The generalization error depends as well on the complexity of the network, which is quantified in terms of the number of edges in the network, the number of time steps, and gradients precision in the gradient descent algorithm. Our first result connects to the SQ-like lower bound presented for the INAL in Section 3.1. Recall the definition of D-extension of a target function f (Definition 7), which we denoted by $\overline{f}$.

Theorem 8. *Consider a fully connected neural network of size P with initialization that is invariant under permutations of the input neurons. Let $f : \{\pm 1\}^d \to \{\pm 1\}$ be a balanced target function. Let $\overline{f}$ be the $2d$-extension of f as defined in Definition 7. Let $\mathrm{NN}(.;\theta^T)$ be the output of noisy-SGD with gradient range A, noise level τ and batch-size B after T time steps. Then, for $\delta < 1/4$ and for B large enough*[IX]*, the generalization error satisfies*

$$|\mathbb{E}_{\mathrm{NN}(.;\theta^T)}\langle \mathrm{NN}(.;\theta^T), \overline{f}\rangle| \leq \frac{T\sqrt{P}A}{\tau} \cdot \mathrm{Stab}_\delta[f]^{1/4}. \tag{3.45}$$

Remark 9.
- *One could consider other groups of invariances than the full permutation group. The analysis of lower bounds for networks invariant to general group actions, depending on the complexity of the target task is left for future work.*
- *Generally, Theorem 8 holds for any δ such that $\mathrm{CP}(\mathrm{orb}(\overline{f})) \leq \mathrm{Stab}_\delta[f]$; in particular, this holds for $\delta < 1/4$ under input doubling. Increasing the magnitude of the input extension allows to increase the upper bound on δ. In Appendix D.2 we discuss how under non-extremal and non-dense assumptions one can remove the input doubling and have Theorem 8 hold for the original d, for some $\delta > 0$.*

Theorem 8 states a lower bound for learning in the extended input space. This is because the proof of Theorem 8 uses Abbe and Sandon, 2020a which obtains a lower-bound in terms of the cross-predictability (CP) (instead of the noise stability), which is a complexity measure of a *class* of functions, rather than a single function (see Definition 8). Similarly to Theorem 4, we use the input doubling to guarantee that the hypothesis class $\mathscr{F}$ resulting from the orbit of $\overline{f}$, i.e., $\mathrm{orb}(\overline{f}) = \{\overline{f} \circ \pi : \pi \in S_{2d}\}$, has large enough cardinality (i.e. not degenerate). Degenerate orbits could indeed be learned using a proper choice of the initialization (e.g. for orbits containing a unique function, one can simply set the weights of the neural network at initialization to represent the unique function, if the network has enough expressivity). Instead, the input doubling prohibits such representation shortcut and ensures that the structural properties of the function is what creates the difficulty of learning, irrespective of the choice of the initialization.

One can remove this input doubling requirement by assuming that f is non-extremal (i.e., no terms of degree $\theta(d)$ in the Fourier basis) and non-dense (i.e., poly(d)-sized Fourier spectrum), see Appendix D.2.

[IX]This holds for $B \geq 1/\mathrm{CP}(\mathrm{orb}(\overline{f}))$.

Theorem 8 states that if the target function is highly noise unstable, specifically if there exists $\delta < 1/4$ such that $\mathrm{Stab}_\delta[f]$ decreases faster than any inverse polynomial in d, then GD will not learn the $2d$-extension of f (or f itself if it is non-extremal/dense) in polynomial time and with a polynomially sized neural network initialized at random. So in that sense, the noise-stability gives a proxy to generalization error, as observed in (Zhang et al., 2021). More specifically, this result is about failure of the weakest form of learning no matter what the architecture is. One could also consider 'regular' architectures (such as with isotropic layers) and stronger learning requirements, but this is left to future work.

More details of the proof of Theorem 8 are in Appendix D.1. In Appendix D.3, we consider Pointer-Value-Retrieval (PVR) tasks, that were mentioned in the Introduction and that will be considered in Chapter 4. We explain how the noise stability of such functions can be computed, and the implications of Theorem 8.

Remark 10. *We remark that with a similar proof technique as the one in Proposition 8 and Theorem 6 and 7 one can remove the input extension trick, by constraining the loss to be the correlation-loss and the architecture to be 2-layer fully connected with* ReLU *activation.*

3.3.1 Noise Stability and Initial Alignment

In this Section, we relate the Noise Stability to the Initial Alignment, introduced in Definition 3. We remark that both noise sensitivity and INAL are related by the cross-predictability (CP). Let us first give two definitions. Recall that for $f : \{\pm 1\}^d \to \{\pm 1\}$, $\mathrm{NS}_\delta[f] = \frac{1}{2} - \frac{1}{2}\mathrm{Stab}_\delta[f]$. Let us recall the definition of "high-degree" functions.

Definition 15 (High-Degree, restatement of Definition 10)**.** *We say that a family of functions $f_d : \{\pm 1\}^d \to \mathbb{R}$ is* high-degree *if for any fixed k, $W^{\leq k}(f_d) = d^{-\omega(1)}$, i.e. $W^{\leq k}(f_d)$ is negligible.*

We further introduce the notion of 'noise sensitive' and 'strongly noise sensitive' functions.

Definition 16 (Noise sensitive function)**.** *We say that a family of functions $f_d : \{\pm 1\}^d \to \{\pm 1\}$ is* noise sensitive *if for any $\delta \in (0, 1/2]$, $\mathrm{NS}_\delta[f_d] = 1/2 - o_d(1)$.*

Definition 17 (Strongly noise sensitive function)**.** *We say that a family on functions $f_n : \{\pm 1\}^d \to \{\pm 1\}$ is* strongly noise sensitive *if for any $\delta \in (0, 1/2]$, $\mathrm{NS}_\delta[f_d] = 1/2 - d^{-\omega(1)}$.*

We then prove the following.

Proposition 9. *Let* $\mathrm{NN}_d : \mathbb{R}^d \to \mathbb{R}$ *be a fully connected neural network with Gaussian i.i.d. initialization and expressive activation (as in Theorem 4). If* $\mathrm{INAL}(\mathrm{NN}_d, f_d) = d^{-\omega(1)}$*, then f_d is noise sensitive.*

Proof. It is enough to show that for any $\delta \in [0, 1/2]$, $\sum_{k=0}^{d} (1-2\delta)^k W^k(f_d) = o_d(1)$, or analogously that for any $\epsilon > 0$ and for d large enough $\sum_{k=0}^{d} (1-2\delta)^k W^k(f_d) < \epsilon$. Fix δ and let

$\epsilon > 0$. Let k_0 be such that $(1-2\delta)^{k_0} < \epsilon/2$. Then,

$$\sum_{k=0}^{d}(1-2\delta)^k W^k(f_d) = \sum_{k=0}^{k_0}(1-2\delta)^k W^k(f_d) + \sum_{k=k_0+1}^{d}(1-2\delta)^k W^k(f_d) \tag{3.46}$$

$$\leq W^{\leq k_0}(f_d) + (1-2\delta)^{k_0+1}\sum_{k=k_0+1}^{d} W^k(f_d). \tag{3.47}$$

By Proposition 3 and Corollary 4, if $\mathrm{INAL}(f_d,\sigma) = d^{-\omega(1)}$ then f_d is high-degree. Thus, $W^{\leq k_0}(f_d) = d^{-\omega(1)}$, and clearly for d large enough $W^{\leq k_0}(f_d) < \epsilon/2$. On the other hand, $\sum_{k=k_0+1}^{d} W^k(f_d) < 1$, since f is Boolean-valued. Thus, $\sum_{k=0}^{d}(1-2\delta)^k W^k(f_d) < \epsilon$, and the Proposition is proven. □

Proposition 10. *If f_d is strongly noise sensitive, then f_d is high-degree.*

Proof. It is enough to show that if $\sum_{k=0}^{d}(1-2\delta)^k W^k(f_d) = d^{-\omega(1)}$ then for any constant k, $W^{\leq k}(f_d) = d^{-\omega(1)}$. Take $k_0 \in \mathbb{N}$, then

$$d^{-\omega(1)} = \sum_{k=0}^{d}(1-2\delta)^k W^k(f_d) \geq \sum_{k=0}^{k_0}(1-2\delta)^k W^k(f_d) \geq (1-2\delta)^{k_0} W^{\leq k_0}(f_d). \tag{3.48}$$

Clearly this implies that $W^{\leq k_0}(f_d) = d^{-\omega(1)}$, and the proof is concluded. □

3.4 Conclusion and Future Work

In this Chapter, we showed lower bounds on the generalization error achieved after training with the noisy-GD algorithm in specific settings, depending on three complexity measures: the initial alignment, the initial gradient alignment and the noise stability.

Regarding the results of Section 3.1, a relevant future work would be to extend the result beyond fully connected architectures. As mentioned in that Section, we suspect that our result can be generalized to all architectures that contain a fully connected layer anywhere in the network. Another direction would be to extend the work to other continuous distributions of initial weights (beyond Gaussian). As a matter of fact, in the setting of iid Gaussian inputs (instead of Boolean inputs), Theorem 4 extends to all weight initialization distributions with zero mean and variance $O(d^{-1})$. However, in the case of Boolean inputs that we consider, this may not be a trivial extension. Another extension, on which we do not touch, are non-uniform input distributions.

In Section 3.2, we address some of the limitations of the result of Section 3.1 (Boolean inputs, necessity of input extension). In particular, we present two lower bounds on the generalization error in term of the 2nd and 4th moment of the ℓ_2 norm of the alignment between the initial gradient and the target function. We choose the initial gradient alignment (instead of the initial alignment), as this is the quantity that arises in our proof technique. However, we believe that for standard activations (e.g. ReLU), the two alignments can be related, and one could express both results in terms of the same measure. There are several future directions to pursue. Our results hold for noisy-GD on the correlation loss (and on the hinge loss, with additional assumption on the initialization). It would be interesting to extend the argument to the square loss, and eventually to general losses. Extending the result to the square loss implies controlling the gradient alignment achieved along the trajectory of the Junk-Flow, which for the square loss reads as:

$$\psi^0 = \theta^0 \tag{3.49}$$

$$\psi^{t+1} = \psi^t - \gamma\left(\mathbb{E}_x[\nabla_{\psi^t} \mathrm{NN}(x;\psi^t)\cdot \mathrm{NN}(x;\psi^t)] + \xi^t\right), \tag{3.50}$$

where ξ^t denotes some Gaussian noise. We believe that a coupling between high degree parities might allow to show that the gradient alignment with the target function stays negligible along (3.50). Moreover, as discussed in Remark 6, it would be interesting to characterize the family of initializations that allow (and do not allow) to learn high degree Boolean functions.

In Section 3.3 we first establish a formal result that supports a conjecture made in (Zhang et al., 2021), relating the noise sensitivity of a target function to the generalization error. This gives a first connection between a central measure in Boolean analysis, the noise-sensitivity, and the generalization error when learning Boolean functions with GD. It would be interesting to generalize the result to a more general setting of architectures, and to show a positive result that relates the generalization error to the noise stability.

4 Mis-Matched Setting

4.1 The Pointer-Value-Retrieval (PVR) benchmark

Recently, (Zhang et al., 2021) introduced the pointer value retrieval (PVR) benchmark. This benchmark consists of a supervised learning task on MNIST (LeCun et al., 2010) digits with a 'logical' or 'reasoning' component in the label generation. More specifically, the functions to be learned are defined on MNIST digits organized either sequentially or on a grid, and the label is generated by applying some 'reasoning' on these digits, with a specific digit acting as a pointer on a subset of other digits from which a logical/Boolean function is computed to generate the label.

For instance, consider the PVR setting for binary digits in the string format, where a string of MNIST digits is used as input. Consider in particular the case where only 0 and 1 digits are used, such as in the example of Figure 4.1. The label of this string is defined as follows: the first 3 bits 101 define the pointer in binary expansion, and the pointer points to a window of a given length, say 2 in this example. Specifically, the pointer points at the first bit of the window. To generate the label, one has to thus look the 6th window[I] of length 2 given by 11, and there the label is produced by applying some fixed function, such as the parity (so the label would be 0 in this example). In (Zhang et al., 2021), the PVR benchmark is also defined for matrices of digits rather than strings; we focus here on the string version that captures all the purposes of the PVR study. This benchmark is introduced to understand the limits of deep learning on tasks that go beyond classical image recognition, investigating in particular the trade-off between memorization and reasoning by acting with a particular distribution shift at testing (see further details below).

In order to learn such PVR functions, one has to first learn the digit identification and then the logical component on these digits. Handling both tasks successfully at once is naturally more demanding than succeeding at the latter assuming the first is successful. One can thus

[I]pointer 000 points at the first window, pointer 001 at the second window, and so on. Thus pointer 101, that is equal to 5 in binary expansion, points at the 6th window.

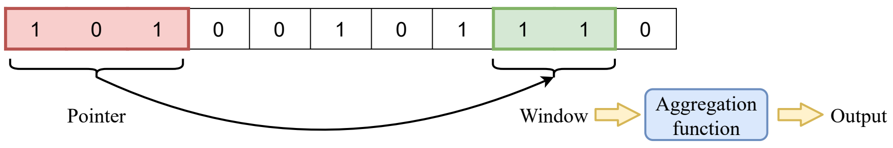

Figure 4.1: An example of a PVR function with a window size of 2. The first 3 bits are the pointer, which points to a window in the subsequent bits. Specifically, the number indicated by the pointer bits in binary expansion gives the position of the first bit of the window. The label is then produced by applying some fixed aggregation function to the window bits (e.g., parity, majority-vote, etc.).

focus on the 'logical component' as a necessary component to learn, and this corresponds to learning a Boolean function. The overall function that maps the pixels of an image to its label in the PVR is of course also a Boolean function (like any computer-encoded function), but the structural properties of such meta-functions are more challenging to describe. In any case, to understand the limits of deep learning on such benchmarks, we focus on investigating first the limits of deep learning on the logical/Boolean component.

We next re-state formally one of the PVR benchmarks from (Zhang et al., 2021), focusing on binary digits for simplicity. We will use the alphabet $\{0,1\}$ to describe the benchmark to connect to the MNIST dataset, but will later switch to the alphabet $\{+1,-1\}$ for the problem of learning Boolean functions with neural networks. Recall for $d \in \mathbb{N}$, $\mathbb{F}_2^d = \{0,1\}^d$.

Definition 18 (Boolean PVR with *sliding* windows)**.** *The input x consists of d bits and the label is generated as*

$$f(x_1,\ldots,x_d) = g(x_{P(x^p)},\ldots,x_{P(x^p)+w-1}). \tag{4.1}$$

where p is the number of bits in the pointer, w is the window size, $g : \mathbb{F}_2^w \to \mathbb{R}$ is the aggregation function, $P : \mathbb{F}_2^p \to [p+1:d]$ is the pointer map, x_j denotes the bit in position j and $x^p = (x_1,...,x_p)$ denotes the pointer bits. We often set $d = p + 2^p$, and hence the last window starts at the last digit.

In words, the first p bits give a pointer to the beginning of a window of w consecutive bits, and the label is produced by applying a Boolean function g on these w bits.

Remark 11. *If $w > 1$, for some values of the pointer, $P(x^p)+w-1$ would exceed the dimension of input n. This issue can be solved by using cyclic indices or by using non-overlapping windows as defined in Appendix D.3. However, in the experiments in Section 4.5, we mainly truncate windows (if necessary) in order to capture the underlying asymmetries (e.g., the last windows have smaller sizes).*

4.2 Canonical Holdout

In this Chapter, we consider the problem of learning in the holdout setting, i.e. when some data are withheld (or 'unseen') during training. We focus on a specific type of holdout setting, that we call 'canonical holdout', which we define here.

Definition 19 (Canonical holdout)**.** *Let $\mathscr{F}$ be the class of Boolean functions on d bits and let f be a specific function in that class. For $k \in [d]$, consider the problem of learning $\mathscr{F}$ from samples $(x_{-k}, f(x_{-k}))$, where the x_{-k}'s are independently drawn from the distribution that freezes component k to 1 and that draws the other components i.i.d. Bernoulli$(1/2)$. Let $\tilde{f}_{-k}$ be the function learned under this training distribution. We are interested in the generalization error with* square loss *when the k-th bit is not frozen at testing, i.e.,*

$$\mathrm{gen}(\mathscr{F}, \tilde{f}_{-k}) = \frac{1}{2}\mathbb{E}_{x\sim_U \mathbb{F}_2^d, f\sim_U \mathscr{F}}\left[(f(x) - \tilde{f}_{-k}(x))^2\right], \tag{4.2}$$

where by $x \sim_U \mathbb{F}_2^d$ we mean that x is chosen uniformly at random from $\mathbb{F}_2^d$, and similarly for $f \sim_U \mathscr{F}$.

Finally, we will consider neural network architectures that are invariant to some transformations on the inputs, such as permutation invariance. In the PVR setting, this means that we do not assume that the learner has knowledge of which bits are in the window. In such cases, instead of learning a class of function $\mathscr{F}$, one can equivalently talk about learning a single function f (with the implicit $\mathscr{F}$ defined as the orbit of f through the invariance group), and define

$$\mathrm{gen}(f, \tilde{f}_{-k}) = \frac{1}{2}\mathbb{E}_{x\sim_U \mathbb{F}_2^d}\left[(f(x) - \tilde{f}_{-k}(x))^2\right].$$

4.3 Boolean Influence and Low-Degree Bias Hypothesis

In this Chapter we put forward the following hypothesis:

> *(S)GD on the square loss*[II] *and on certain network architectures such as MLPs and Transformers has an implicit bias towards low-degree representations when learning logical functions such as Boolean PVR functions.*

Let us give an illustration of this hypothesis. Before starting, recall that any Boolean function $f : \{\pm 1\}^d \to \mathbb{R}$ can be written in terms of its Fourier-Walsh transform:

$$f(x) = \sum_{T\subseteq[d]} \hat{f}(T)\chi_T(x), \tag{4.3}$$

[II]We do not expect the square loss to be critical, but it makes the connection to the Boolean influence more explicit.

where $\chi_T(x) = \prod_{i\in T} x_i$ and $\hat{f}(T) = \mathbb{E}_x[f(x)\chi_T(x)]$ are respectively the basis elements and the coefficients of the Fourier-Walsh transform of f. The Boolean influence is defined as follows.

Definition 20 (Boolean influence (O'Donnell, 2014)). *Let $f : \{\pm 1\}^d \to \mathbb{R}$ be a Boolean function and let $\hat{f}$ be its Fourier-Walsh transform. The Boolean influence of variable $k \in [d]$ on f is defined by*

$$\mathrm{Inf}_k(f) := \sum_{T\subseteq[d]:k\in T} \hat{f}(T)^2. \tag{4.4}$$

In particular, if $f : \{\pm 1\}^d \to \{\pm 1\}$,

$$\mathrm{Inf}_k(f) = \mathbb{P}(f(x) \neq f(x+e_k)), \tag{4.5}$$

where $x + e_k$ corresponds to the vector obtained by flipping the k-th component of x.

Example 6. *Assume $x \sim \mathrm{Unif}\{\pm 1\}^d$.*

- *(Parities): For $\chi_S(x) := \prod_{i\in S} x_i$, for $S \subseteq [d]$:*

$$\mathrm{Inf}_k(\chi_S) = \mathbb{1}(k \in S). \tag{4.6}$$

  Indeed, flipping any coordinates in S flips the output of χ_S with probability 1, whereas the output of χ_S is not affected by changes in coordinates outside its support.

- *(Majority): If $\mathrm{Maj}_d(x) = \mathrm{sgn}(\sum_{i=1}^d x_i)$, with d odd:*

$$\mathrm{Inf}_k\left(\mathrm{Maj}_d\right) = 2^{-(d-1)} \binom{d-1}{\frac{d-1}{2}} \quad \forall k \in [d]. \tag{4.7}$$

  Indeed, flipping coordinate k matters if and only if exactly $\frac{d-1}{2}$ other bits have the same sign as the k-th bit.

Consider the following example of a PVR function with a 1-bit pointer, 2 overlapping windows of length 2, and parity for the aggregation function. We consider $f : \{\pm 1\}^d \to \{\pm 1\}$ (i.e., with ± 1 variables instead of 0, 1, to simplify the expressions), with f given by

$$f(x_1,x_2,x_3,x_4) = \frac{1+x_1}{2} x_2x_3 + \frac{1-x_1}{2} x_3x_4. \tag{4.8}$$

We can rewrite f in terms of its Fourier-Walsh expansion (i.e., pulling out all the multivariate monomials appearing in the function, see Section 4.5 for more details), which gives

$$f(x_1,x_2,x_3,x_4) = \frac{1}{2}x_2x_3 + \frac{1}{2}x_3x_4 + \frac{1}{2}x_1x_2x_3 - \frac{1}{2}x_1x_3x_4. \tag{4.9}$$

Consider now training a neural network such as a Transformer as in Section 4.5 on this function, with quadratic loss, and a canonical holdout corresponding to freezing $x_2 = 1$ at training. Under this holdout, and under the 'low-degree implicit bias' hypothesis, the low-degree monomials are learned first (see experiments in Section 4.5), resulting in the following function being learned at training:

$$f_{-2}(x_1, x_2, x_3, x_4) = \frac{1}{2}x_3 + \frac{1}{2}x_3x_4 + \frac{1}{2}x_1x_3 - \frac{1}{2}x_1x_3x_4 = \frac{1+x_1}{2}x_3 + \frac{1-x_1}{2}x_3x_4. \tag{4.10}$$

Thus, according to Lemma 2 proved in Appendix E, the generalization error is given by

$$\frac{1}{2}\mathbb{E}(f(x) - f_{-2}(x))^2 = \mathbb{P}(f(x) \neq f(x+e_2)) = \frac{1}{2}, \tag{4.11}$$

which is the probability that flipping the frozen coordinate changes the value of the target function, i.e., the Boolean influence (where we denoted by $X + e_2$ the vector obtained by flipping the second entry in X). As shown in this Chapter, neural networks tend to follow this trend quite closely, and we can prove this hypothesis on simple linear models. Notice that an ERM-minimizing function could have taken a more general form than a degree minimizing function, i.e.,

$$f_{-2}^{\text{ERM}}(x) := \frac{1+x_2}{2}f_{-2}(x) + \frac{1-x_2}{2}r(x) \tag{4.12}$$

for any choice of $r : \{\pm1\}^4 \to \{\pm1\}$. In the special case of $r = 0$, f_{-2}^{ERM} corresponds to f_{-2} (the low-degree representation).

For instance, among such ERM-minimizers, one can check that the minimum ℓ_2-norm interpolating solution would be given by

$$f_{-2}^{\ell_2}(x) := \frac{1}{4}(x_3 + x_2x_3) + \frac{1}{4}(x_3x_4 + x_2x_3x_4) + \frac{1}{4}(x_1x_3 + x_1x_2x_3) - \frac{1}{4}(x_1x_3x_4 + x_1x_2x_3x_4). \tag{4.13}$$

This gives a generalization error of $\frac{1}{2}\mathbb{E}(f(x) - f_{-2}^{\ell_2}(x))^2 = 4(\frac{1}{4})^2 = \frac{1}{4}$, i.e., half the error of f_{-2}, yet still bounded away from 0.

In order to improve on this, under the same canonical holdout with $x_2 = 1$, one would like to rely on a type of minimum description length bias, since describing f may be more efficient than f_{-2} due to the stronger symmetries of f. Namely, f corresponds to taking the parity on the middle two bits if $x_1 = 1$, and on the last two bits otherwise. On the other hand, f_{-2} requires changing the function depending on $x_1 = 1$ or $x_1 = -1$, since it is once the function x_3 and once the function x_3x_4. So an implicit bias that would exploit such symmetries, featuring in PVR tasks, would give a different solution than the low-degree implicit bias, and could result in lower generalization error. We leave to future work to investigate this 'symmetry compensation' procedure.

4.4 Related Literature

Implicit bias. The implicit bias of neural networks trained with gradient descent has been extensively studied in recent years (Gunasekar et al., 2018a; Ji and Telgarsky, 2019; Moroshko et al., 2020; Neyshabur et al., 2017; Neyshabur et al., 2014). In particular, (Soudry et al., 2017) proved that gradient descent on linearly-separable binary classification problems converges to the maximum ℓ_2-margin direction. Several subsequent works studied the implicit bias in classification tasks on various networks architectures, e.g., homogeneous networks (Lyu and Li, 2019), two layers networks in the mean field regime (Chizat and Bach, 2020), linear and ReLU fully connected networks (Vardi et al., 2021), and convolutional linear networks (Gunasekar et al., 2018b). Among regression tasks, the problem of implicit bias has been analysed for matrix factorization tasks (Arora et al., 2019; Gunasekar et al., 2017; Razin and Cohen, 2020), and also gradient flow (Berthier, 2022; Pesme et al., 2021) and (S)GD (Even et al., 2023) on diagonal linear networks. However, all these works consider functions with real inputs, instead of logical functions which are the focus of this work. On the other hand, as discussed in Section 4.3, the Boolean influence generalization characterization reflects the implicit bias of GD on neural networks to learn low-degree representations. Similar types of phenomena can implicitly be found in (Rahaman et al., 2019; Xu et al., 2019; Xu et al., 2018), in particular as the "spectral bias" in the context of real valued functions decomposed in the classical Fourier basis (where the notion of lower degree is replaced by low frequencies). In (Abbe, Boix-Adsera, et al., 2023; Abbe, Boix-Adserà, et al., 2021; Abbe, Boix-Adserà, and Misiakiewicz, 2022), the case of Boolean functions is considered as in this Thesis, and it is established that for various 'regular' architectures (having some symmetry in their layers), gradient descent can learn target functions that satisfy a certain 'staircase' property.

Distribution shift. Many works were aimed at characterizing when a classifier trained on a training distribution (also called the "source" distribution) performs well on a different test domain (also called the "target" distribution) (Quionero-Candela et al., 2009). On the theoretical side, (Ben-David et al., 2010) obtains a bound of the target error of a general empirical risk minimization algorithm in terms of the source error and the divergence between the source and target distribution, in a setting where the algorithm has access to a large dataset from the source distribution and few samples from the target distribution. We refer to (Shen et al., 2017) for a further result in a similar setting. Instead, in this work we focus on gradient descent on neural networks in the setting where no data from the target distribution is accessible. On the empirical level, several benchmarks have been proposed to evaluate performance for a wide range of models and distribution shifts (Miller et al., 2021; Sagawa et al., 2021; Wiles et al., 2022). Despite this significant body of works on distribution shift, we did not find works that related the generalization error under holdout shift in terms of the Boolean influence.

4.5 Results

Recall, the canonical holdout setting defined in Definition 19: component k is frozen to 1 during training, and it is released to Unif{±1} at testing. Our experiments show that in this setting, for some relevant architectures, the generalization error is close to the Boolean influence of variable k on f (Definition 20).

4.5.1 Boolean influence and generalization

To explain the connection between generalization error and Boolean influence, we start with a simple lemma relating the Boolean influence to the ℓ_2-distance between the true target function and the function obtained by freezing component k (which we call the "frozen function").

Lemma 2. *Let $f : \{\pm 1\}^d \to \mathbb{R}$ be a Boolean function and let f_{-k} be defined as $f_{-k}(x) := f(x_{-k})$ where $x_{-k}(i) = 1$ if $i = k$ and $x_{-k}(i) = x(i)$ otherwise. Then,*

$$\frac{1}{2}\mathbb{E}_{x\sim\mathrm{Unif}\{\pm1\}^d}\left[(f(x)-f_{-k}(x))^2\right] = \mathrm{Inf}_k(f). \tag{4.14}$$

The proof of Lemma 2 can be found in Appendix E. In Section 4.5.2, we present experiments that demonstrate the relation between the Boolean influence and the generalization error for different architectures. In Section 4.5.3, we focus on linear models.

4.5.2 Experiments

We consider three architectures for our experiments: multi-layer perceptron (MLP) with 4 hidden layers, the Transformer (Vaswani et al., 2017), and MLP-Mixer (Tolstikhin et al., 2021). For each architecture, we train the model while freezing coordinate 1, then coordinate 2 and so on, until coordinate d, and compare the generalization error with the Boolean influence of the corresponding coordinates of the target function. We train our models using ℓ_2 loss and mini-batch SGD with momentum and batch-size of 64 as the optimizer. Moreover, we have repeated each experiment 40 times and averaged the results. Furthermore, note that for the MLP model, we pass the Boolean vector directly to the model. However, for the Transformer and MLP-Mixer, we first encode $+1$ and -1 tokens into 256 dimensional vectors and then we pass the embedded input to the models. More details on training procedure as well as further experiments can be found in (Abbe, Bengio, et al., 2022, Appendix F).

Influence v.s. canonical holdout generalization. In this section, we consider a Boolean PVR function (as in Def. 18) with 3 pointer bits to be learned by the neural networks. For this function we set window size to 3 and use majority-vote (defined as $g(x_1,\dots,x_r) = \mathrm{sign}(x_1+\cdots+x_r)$, outputting -1, 0, or 1) as the aggregation function on the windows. The generalization

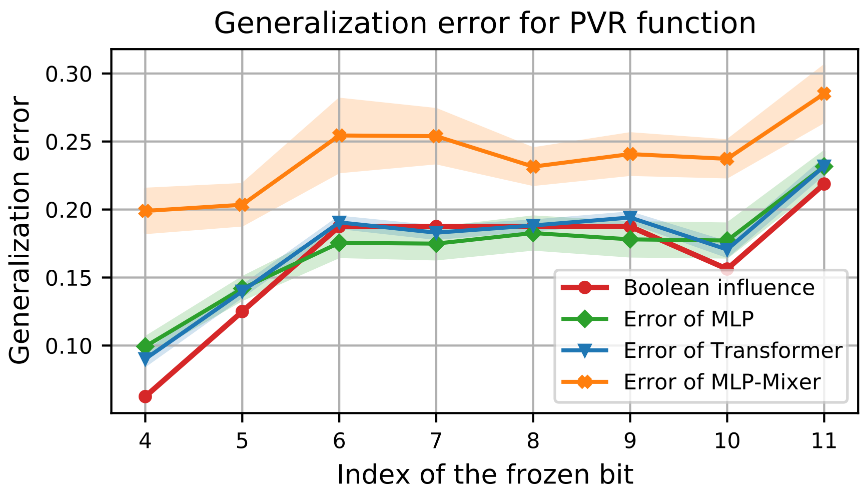

Figure 4.2: Comparison between the generalization loss in the canonical distribution shift setting and the Boolean influence for a PVR function with 3 pointer bits, window size 3, and majority-vote aggregation function.

error of the models on this PVR function and its comparison with the Boolean influence are presented in Figure 4.2. The x-axis corresponds to the index of the frozen coordinate, that is from 4 to 11 (we do not freeze the pointer bits). On the y-axis, for each frozen coordinate we report the generalization error obtained in the corresponding holdout setting for MLP, the Transformer, and MLP-Mixer, together with the value of the Boolean influence of the frozen coordinate on the target function. It can be seen that the generalization error of MLP and the Transformer can be well approximated by the Boolean influence. Whereas, the generalization error of MLP-Mixer follows the trend of the Boolean influence with an offset. We remark that in Figure 4.2, the value of the Boolean influence (and gen. error) in the PVR task varies across different indices due to boundary effects and the use of truncated windows (see Def. 18). We refer to (Abbe, Bengio, et al., 2022, Appendix F) for further experiments on other target functions.

Implicit bias towards low-degree representation. Consider the problem of learning a function f in the canonical holdout setting freezing $x_k = 1$. Denote the Fourier coefficients of the frozen function f_{-k} (as defined in Lemma 2), by $\hat{f}_{-k}(S)$, for all $S \subseteq [d] := \{1, ..., d\}$ (recall, $\hat{f}_{-k}(S) = \mathbb{E}_x[f_{-k}(x)\chi_S(x)]$). For S such that $k \notin S$, the neural network can learn coefficient $\hat{f}_{-k}(S)$ using either $\chi_S(x)$ or $x_k \cdot \chi_S(x) = \chi_{S\cup\{k\}}(x)$ (since these are indistinguishable at training). The low-degree implicit bias states that neural networks have a preference for the lower degree monomial, i.e., χ_S. According to Lemma 2, the generalization error will be close to the Boolean influence if χ_S is learned faster than $\chi_{S\cup\{k\}}$ and thus the term $\hat{f}_{-k}(S)\chi_S(x)$ in the Fourier expansion of f_{-k}, is mostly learned by the lower degree monomial. Figure 4.3 shows this bias empirically for the above mentioned PVR function and for frozen coordinate $k = 6$. Figure 4.3 (top-left) shows that the MLP model has a strong preference for low-degree

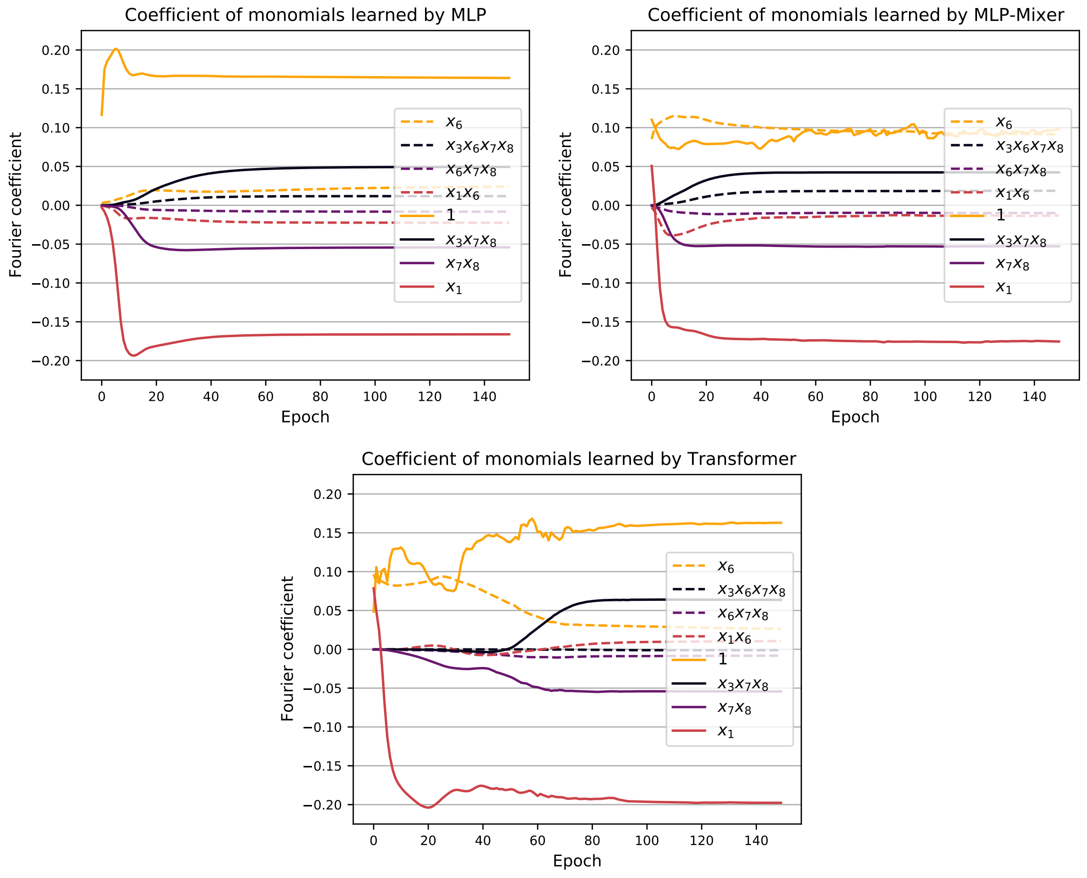

Figure 4.3: The coefficients of a selected group of monomials learned by the MLP (top-left), MLP-Mixer (top-right) and the Transformer (bottom) when learning the aforementioned PVR function, with $x_6 = 1$ frozen during the training. The coefficient of these monomials in the original function are $\hat{f}(\{6\}) = 0.1875$, $\hat{f}(\{3,6,7,8\}) = 0.0625$, $\hat{f}(\{6,7,8\}) = -0.0625$, and $\hat{f}(\{1,6\}) = -0.1875$. One can observe that the monomials of the lowest degree are indeed picked up first during the training of the MLP and Transformer, which explains the tight approximation of the Boolean influence for the generalization error in these cases. In contrast, the MLP-Mixer also picks up some contribution from the higher degree monomials including the frozen bit x_6.

monomials (continuous lines), i.e., it captures the monomials in the original function using monomials that exclude the frozen index. Therefore the generalization error of MLP is very close to the Boolean influence as seen in Figure 4.2. Whereas, Figure 4.3 (top-right) shows that the MLP-Mixer model has a weaker preference for lower degree monomials and hence, its generalization error follows the trend of Boolean influence with an offset, which is also present in Figure 4.2.

4.5.3 Linear Regression Models

In this section, we will focus on linear functions and we state a theorem for linear regression models.

Theorem 9. *Let $f : \{\pm 1\}^d \to \mathbb{R}$ be a linear function, i.e., $f(x_1, \cdots, x_d) = \hat{f}(\emptyset) + \sum_{i=1}^d \hat{f}(\{i\})x_i$. Consider the canonical holdout where the k-th component is frozen at training for a linear regression model where weights and biases are initialized independently with the same mean and variance σ^2. Also assume the frozen function is unbiased, i.e., $\mathbb{E}_{x_{-k}}[f(x)] = 0$. In this case, the expected generalization error (over different initializations) of the function learned by GD after t time steps is given by*

$$\mathbb{E}_{\theta^0}[\text{gen}(f, \tilde{f}_{-k}^{(t)})] = \text{Inf}_k(f) + o_{\sigma^2}(1) + O(e^{-ct}), \tag{4.15}$$

where c is a constant dependent on the learning rate. Moreover, if the frozen function is biased, the expected generalization error is equal to

$$\mathbb{E}_{\theta^0}[\text{gen}(f, \tilde{f}_{-k}^{(t)})] = \frac{(\hat{f}(\emptyset) - \hat{f}(\{k\}))^2}{4} + o_{\sigma^2}(1) + O(e^{-ct}). \tag{4.16}$$

The proof of Theorem 9 is presented in Appendix E.

4.6 Conclusion

In this Chapter we investigate the generalization error under the canonical holdout. The 'low-degree implicit bias hypothesis' is put forward and supported both experimentally and theoretically for certain architectures. This gives a new insight on the implicit bias of GD when training neural networks, that is specific to Boolean functions such as the Boolean PVR, and that relates to the fact that certain networks tend to greedily learn monomials by incrementing their degree. In particular, this allows to characterize the generalization error in terms of the Boolean influence, a second central notion in Boolean Fourier analysis. Boolean measures thus seem to have a role to play in understanding generalization when learning of 'reasoning' or 'logical' functions. There are many directions to pursue such as:

1. Extending the realm of architectures/models for which we can prove formally the Boolean influence tightness.

2. Considering more general holdout or distribution shift models. In (Abbe, Bengio, et al., 2023) the authors consider a stronger case of out-of distribution generalization and show convergence to a *min-degree-interpolator* for several models, including Transformers, random features models and diagonal linear networks.

3. Investigating how the picture changes when the bits/digits are given by MNIST images.

4. Better understanding when the low-degree implicit bias is taking place or not, within and beyond PVR, since the Boolean influence is not always tight in our experiments (e.g., MLP-Mixers seem to have a worse performance than the Boolean influence on PVR).

5. Investigating how to 'revert' the implicit bias towards low-degree when it is taking place, to compensate for the unseen data; this will require justifying and engineering why certain symmetries are favorable in the learned function.

5 Curriculum Learning for Parities

5.1 Introduction

Several experimental studies have shown that humans and animals learn considerably better if the learning materials are presented in a curated, rather than random, order (Avrahami et al., 1997; Elio and Anderson, 1984; Ross and Kennedy, 1990; Shafto et al., 2014). This is broadly reflected in the educational system of our society, where learning is guided by an highly organized curriculum. This may involve several learning steps: with easy concepts introduced at first and harder concepts built from previous stages.

Inspired by this, (Bengio et al., 2009) formalized a *curriculum learning (CL)* paradigm in the context of machine learning and showed that for various learning tasks it provided improvements in both the training speed and the performance obtained at convergence. This seminal paper inspired many subsequent works, that studied curriculum learning strategies in various application domains, e.g. computer vision (Dong et al., 2017; Sarafianos et al., 2017), computational biology (Xiong et al., 2021), auto-ML (Graves et al., 2017), natural language modelling (Campos, 2021; Shi et al., 2013, 2015; Zaremba and Sutskever, 2014). While extensive empirical analysis of CL strategies have been carried out, there is a lack of theoretical analysis. In this Chapter, we make progress in this direction.

A stylized family of functions that is known to pose computational barriers is the class of k-parities over d bits of a binary string. In this Chapter we focus on this class. To define this class: for each subset S of coordinates, the parity over S is defined as +1 if the number of negative bits in S is even, and −1 otherwise, i.e. $\chi_S(x) := \prod_{i \in S} x_i$, $x_i \in \{\pm 1\}$. The class of k-parities contains all χ_S such that $|S| = k$ and it has cardinality $\binom{d}{k}$. Learning k-parities requires learning the support of χ_S by observing samples $(x, \chi_S(x))$, $x \in \{\pm 1\}^d$, with the knowledge of the cardinality of S being k. This requires finding the right target function among the $\binom{d}{k}$ functions belonging to the class.

Learning parities is always possible, and efficiently so, by specialized methods (e.g. Gaussian elimination over the field of two elements). Moreover, (Abbe and Sandon, 2020a) shows that there exists a neural net that learns parities of any degree if trained by SGD with small batch size. However, this is a rather unconventional net. In fact, under the uniform distribution, parities are not efficiently learnable by population queries with any polynomially small noise. The latter can be explained as follows. Assume we sample our binary string uniformly at random, i.e. for each $i \in \{1,...,d\}$, $x_i \sim \mathrm{Rad}(1/2)$[1]. Then, the covariance between two parities $\chi_S, \chi_{S'}$ is given by:

$$\mathbb{E}_{x\sim\mathrm{Rad}(1/2)^{\otimes d}}\left[\chi_S(x)\chi_{S'}(x)\right] = \begin{cases} 1 & \text{if } S = S', \\ 0 & \text{if } S \neq S', \end{cases}$$

where $x \sim \mathrm{Rad}(1/2)^{\otimes d}$ denotes the product measure such that $x_i \overset{iid}{\sim} \mathrm{Rad}(1/2)$, $i \in \{1,...,d\}$. More abstractly, a parity function of k bits is uncorrelated with *any* function of $k-1$ or less bits. This property makes parities hard to learn for any progressive algorithm, such as gradient descent. Indeed, when trying to learn the set of relevant features, a learner cannot know how close its progressive guesses are to the true set. In other words, all wrong guesses are indistinguishable, which suggests that the learner might have to perform exhaustive search among all the $\binom{d}{k}$ sets.

The hardness of learning unbiased parities - and more in general any classes of functions with low cross-correlations - with gradient descent has been analysed e.g. in (Abbe and Sandon, 2020a), where the authors show a lower bound on the computational complexity of learning low cross-correlated classes with gradient-based algorithms with bounded gradient precision. For k-parities, this gives a computational lower bound of $d^{\Omega(k)}$ for any architecture and initialization.

However, if we look at different product distributions, then the inner product of a monomial and a component x_i that is inside and outside the support becomes distinguishable. Suppose the inputs are generated as $x \sim \mathrm{Rad}(p)^{\otimes d}$, for some $p \in (0,1)$. Then the covariance between χ_S and $\chi_{S'}$ is:

$$\begin{aligned}\mathbb{E}_{x\sim\mathrm{Rad}(p)^{\otimes d}}&\Big[(\chi_S(x) - \mathbb{E}[\chi_S(x)])\cdot(\chi_{S'}(x) - \mathbb{E}[\chi_{S'}(x)])\Big] \\ &= \mu_p^{2k-|S\cap S'|} - \mu_p^{2k},\end{aligned}$$

where we denoted by $\mu_p := \mathbb{E}_{z\sim\mathrm{Rad}(p)}[z] = 2p-1$. This implies that if for instance $|p-0.5| > 0.1$, just computing correlations with each bit, will recover the parity with complexity linear in d and exponential in k. If we choose $p = 1 - 1/k$, say, we can get a complexity that is linear in d and polynomial in k. Moreover, the statements above hold even for parities with random noise.

This may lead one to believe that learning biased parities is easy for gradient descent based methods for deep nets. Indeed, (Malach et al., 2021) showed that biased parities are learnable

[1] $z \sim \mathrm{Rad}(p)$ if $\mathbb{P}(z=1) = 1 - \mathbb{P}(z=-1) = p$.

by SGD on a differentiable model consisting of a linear predictor and a fixed module implementing the parity. However, if we consider fully connected networks, as our experiments show (Figure 5.1), while gradient descent for a p far from a half converges efficiently to zero training loss, the learned function actually has *non-negligible error* when computed with respect to the uniform measure. This is intuitively related to the fact that, by concentration of measure, there are essentially no examples with Hamming weight[II] close to $d/2$ in the training set sampled under $\mathrm{Rad}(p)^{\otimes d}$, and therefore it is not reasonable to expect for a general algorithm like gradient descent on fully connected networks (that does not know that the target function is a parity) to learn the value of the function on such inputs.

We thus propose a more subtle question: Is it possible to generate examples from different product distributions and present them in a specific order, in such a way that the error with respect to the unbiased measure becomes negligible?

As we mentioned, training on examples sampled from a biased measure is not sufficient to learn the parity under the unbiased measure. However, it does identify the support of the parity. Our curriculum learning strategy is the following: We initially train on inputs sampled from $\mathrm{Rad}(p)^{\otimes d}$ with p close to 1, then we move (either gradually or by sharp steps) towards the unbiased distribution $\mathrm{Rad}(1/2)^{\otimes d}$. We show that this strategy allows to learn the k-parity problem with a computational cost of $d^{O(1)}$ with SGD on the hinge loss or on the *covariance loss* (see Def. 25). In our proof, we consider layer-wise training (similarly to e.g. (Barak et al., 2022)) and the result is valid for any (even) k and d.

As we mentioned earlier, the failure of learning parities under the uniform distribution from samples coming from a different product measure is due to concentration of Hamming weight. This leads us to consider a family of functions that we call *Hamming mixtures.* Given an input x, the output of a Hamming mixture is a parity of a subset S of the coordinates, where the subset S depends on the Hamming weight of x (see Def. 23). Our intuition is based on the fact that given a polynomial number of samples from, say, the $p = 1/4$ biased measure, it is impossible to distinguish between a certain parity χ_S and a function that is χ_S, for x's whose Hamming weight is at most $3/8d$, and a different function χ_T, for x's whose Hamming weight is more than $3/8d$, for some T that is disjoint from S. In other words, a general algorithm does not know whether there is consistency between x's with different Hamming weight. We show a lower bound for learning Hamming mixtures with curriculum strategies that do not allow to get enough samples with relevant Hamming weight.

Of course, curriculum learning strategies with enough learning steps allow to obtain samples from several product distributions, and thus with all relevant Hamming weights. Therefore, we expect that CL strategies with unboundedly many learning steps will be able to learn the Hamming mixtures.

While our results are restricted to a limited and stylized setting, we believe they may open

[II]The Hamming weight of $x \in \{\pm 1\}^d$ is: $H(x) = \sum_{i=1}^{d} \mathbb{1}(x_i = 1)$.

new research directions. Indeed, we believe that our general idea of introducing correlation among subsets of the input coordinates to facilitate learning, may apply to more general settings. We discuss some of these future directions in the conclusion section of the Chapter.

5.1.1 Related Work

Learning Parities on Uniform Inputs. Learning k-parities over d bits requires determining the set of relevant features among $\binom{d}{k}$ possible sets. The statistical complexity of this problem is thus $\theta(k\log(d))$. The computational complexity is harder to determine. k-parities can be solved in $d^{O(1)}$ time by specialized algorithms (e.g. Gaussian elimination) that have access to at least d samples. In the statistical query (SQ) framework (Kearns, 1998) - i.e. when the learner has access only to noisy queries over the input distribution - k-parities cannot be learned in less then $\Omega(d^k)$ computations. (Abbe and Sandon, 2020a; Shalev-Shwartz et al., 2017) showed that gradient-based methods suffer from the same SQ computational lower bound if the gradient precision is not good enough. On the other hand, (Abbe and Sandon, 2020a) showed that one can construct a very specific network architecture and initialization that can learn parities beyond this limit. This architecture is however far from the architectures used in practice. (Barak et al., 2022) showed that SGD can learn sparse k-parities with SGD with batch size $d^{\theta(k)}$ on a small network. Moreover, they empirically provide evidence of 'hidden progress' during training, ruling out the hypothesis of SGD doing random search. (Andoni et al., 2014) showed that parities are learnable by a $d^{\theta(k)}$ network. The problem of learning *noisy* parities (even with small noise) is conjectured to be intrinsically computationally hard, even beyond SQ models (Alekhnovich, 2003).

Learning Parities on Non-Uniform Inputs. Several works showed that when the input distribution is not the Unif$\{\pm 1\}^d$, then neural networks trained by gradient-based methods can efficiently learn parities. (Malach et al., 2021) showed that biased parities are learnable by SGD on a differentiable model consisting of a linear predictor and fixed module implementing the parity. (Daniely and Malach, 2020) showed that sparse parities are learnable on a two layers network if the input coordinates outside the support of the parity are uniformly sampled and the coordinates inside the support are correlated. To the best of our knowledge, none of these works propose a curriculum learning model to learn parities under the uniform distribution.

Curriculum Learning. *Curriculum Learning (CL)* in the context of machine learning has been extensively analysed from the empirical point of view (Bengio et al., 2009; Soviany et al., 2022; Wang et al., 2021). However, theoretical works on CL seem to be more scarce. In (Saglietti et al., 2022) the authors propose an analytical model for CL for functions depending on a sparse set of relevant features. In their model, easy samples have low variance on the irrelevant features, while hard samples have large variance on the irrelevant features. In contrast, our model does not require knowledge of the target task to select easy examples. In (Weinshall and Amir, 2020; Weinshall et al., 2018) the authors analyse curriculum learning strategies in convex models and show an improvement on the speed of convergence of SGD. In contrast, our work covers an intrinsically non-convex problem. Some works also analysed variants of CL: e.g. self-paced CL (SPCL), i.e. curriculum is determined by both prior knowledge and the training process (Jiang et al., 2015), implicit curriculum, i.e. neural networks tend to consistently learn the samples in a certain order (Toneva et al., 2018). Another form of guided learning appears in (Abbe, Boix-Adsera, et al., 2023; Abbe, Boix-Adserà, et al., 2021; Abbe, Boix-Adserà, and Misiakiewicz, 2022), where the authors analyse staircase functions - sum of nested monomials of increasing degree - and show that the hierarchical structure of such tasks guides SGD to learn high degree monomials. Furthermore, (Kalimeris et al., 2019; Refinetti et al., 2022) show that SGD learns functions of increasing complexity during training. In a concurrent work (Abbe, Bengio, et al., 2023), the authors propose a curriculum learning algorithm (named 'Degree Curriculum') that consists of training on Boolean inputs of increasing Hamming weight, and they empirically show that it reduces the sample complexity of learning parities on small input dimension.

5.2 Definitions and Main Results

We define a curriculum strategy for learning a general Boolean target function. We will subsequently restrict our attention to the problem of learning parities or mixtures of parities. For brevity, we denote $[d]=\{1,...,d\}$. Assume that the network is presented with samples $(x,f(x))$, where $x\in\{\pm1\}^d$ is a Boolean vector and $f:\{\pm1\}^d\to\mathbb{R}$ is a target function that generates the labels. We consider a neural network $\mathrm{NN}(x;\theta)$, whose parameters are initialized at random from an initial distribution P_0, and trained by stochastic gradient descent (SGD) algorithm, defined by:

$$\theta^{t+1}=\theta^t-\gamma_t\frac{1}{B}\sum_{i=1}^{B}\nabla_{\theta^t}L(\theta^t,f,x_i^t), \tag{5.1}$$

for all $t\in\{0,...,T-1\}$, where L is an almost surely differentiable loss-function, γ_t is the learning rate, B is the batch size and T is the total number of training steps. For brevity, we write $L(\theta^t,f,x):=L(\mathrm{NN}(.;\theta^t),f,x)$. We assume that for all $i\in[B]$, $x_i^t\overset{iid}{\sim}\mathcal{D}^t$, where $\mathcal{D}^t$ is a step-dependent input distribution supported on $\{\pm1\}^d$. We define our curriculum learning strategy as follows. Recall that $z\sim\mathrm{Rad}(p)$ if $\mathbb{P}(z=1)=1-\mathbb{P}(z=-1)=p$.

Definition 21 (r-steps curriculum learning (r-CL)). *For a fixed $r\in\mathbb{N}$, let $T_1,...T_r\in\mathbb{N}$ and $p_1,...,p_r\in[0,1]$. Denote by $\bar{p}:=(p_1,...,p_r)$ and $\bar{T}:=(T_1,...,T_{r-1})$. We say that a neural network*

$\mathrm{NN}(x;\theta^t)$ *is trained by SGD with a r-CL*$(\bar{T},\bar{p})$ *if* θ^t *follows the iterations in* (5.1) *with:*

$$\begin{aligned}
&\mathscr{D}^t = \mathrm{Rad}(p_1), && 0 < t \le T_1,\\
&\mathscr{D}^t = \mathrm{Rad}(p_2), && T_1 < t \le T_2,\\
&\dots\\
&\mathscr{D}^t = \mathrm{Rad}(p_r), && T_{r-1} < t \le T.
\end{aligned}$$

We say that r is the number of curriculum steps.

We assume r to be independent on T, in order to distinguish the r-CL from the *continuous*-CL (see Def. 24 below). We hypothesize that r-CL may help to learn several Boolean functions, if one chooses appropriate r and $\bar{p}$. However, in this Chapter we focus on the problem of learning unbiased k-parities. For such class, we obtained that choosing $r = 2$, a wise $p_1 \in (0,1/2)$ and $p_2 = 1/2$ is enough to learn the target parity in $d^{O(1)}$ steps. An interesting future direction would be studying the optimal r and $\bar{p}$. Before stating our Theorem, let us clarify the generalization error that we are interested in. As mentioned before, we are interested in learning the target over the uniform input distribution.

Definition 22 (Generalization error)**.** *We say that SGD on a neural network* $\mathrm{NN}(x;\theta)$ *learns a target function* $f:\{\pm1\}^d \to \mathbb{R}$ *with r-CL$(\bar{T},\bar{p})$ up to error ϵ, if it outputs a network* $\mathrm{NN}(x;\theta^T)$ *such that:*

$$\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[L(\theta^T, f, x)\right] \le \epsilon, \tag{5.2}$$

where L is any loss function such that $\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}[L(f,f,x)] = 0$.

We state here our main theoretical result informally. We refer to Section 5.3.1 for the formal statement with exact exponents and remarks.

Theorem 10 (Main positive result, informal)**.** *There exists a 2-CL strategy such that a 2-layer fully connected network of $d^{O(1)}$ size trained by SGD with batch size $d^{O(1)}$ can learn any k-parities (for k even) up to error ϵ in at most $d^{O(1)}/\epsilon^2$ iterations.*

Let us analyse the computational complexity of the above. At each step, the number of computations performed by a 2-layer fully connected network is given by:

$$(dN + N)\cdot B, \tag{5.3}$$

where d is the input size, N is the number of hidden neurons and B is the batch size. Multiplying by the total number of steps and substituting the bounds from the Theorem we get that we can learn the k-parity problem with a 2-CL strategy in at most $d^{O(1)}$ total computations. Specifically, $O(1)$ denotes quantities that do not grow with k or d, and the statement holds also for large k, d. We prove the Theorem in two slightly different settings, see Section 5.3.1.

One may ask whether the r-CL strategy is beneficial for learning general target tasks (i.e. beyond parities). While we do not have a complete picture to answer this question, we propose a class of functions for which some r-CL strategies are not beneficial. We call those functions the *Hamming mixtures*, and we define them as follows.

Definition 23 ((S,T,ϵ)-Hamming mixture)**.** *For $\epsilon \in [0,1]$, $S, T \in [d]$, we say that $G_{S,T,\epsilon} : \{\pm 1\}^d \to \mathbb{R}$ is a (S,T,ϵ)-Hamming mixture if*

$$G_{S,T,\epsilon}(x) := \chi_S(x)\mathbb{1}(H(x) \le \epsilon d) + \chi_T(x)\mathbb{1}(H(x) > \epsilon d),$$

where $H(x) := \sum_{i=1}^{d} \mathbb{1}(x_i = 1)$ is the Hamming weight of x, $\chi_S(x) := \prod_{i \in S} x_i$ and $\chi_T(x) := \prod_{i \in T} x_i$ are the parity functions over set S and T respectively.

We will consider $\epsilon \neq 1/2$. The intuition of why such functions are hard for some r-CL strategies is the following. Assume we train on samples $(x, G_{S,T.\epsilon}(x))$, with S, T disjoint and $\epsilon \in (0, 1/2)$. Assume that we use a 2-CL strategy and we initially train on samples $x \sim \text{Rad}(p)^{\otimes d}$ for some $p < \epsilon$. If the input dimension d is large, then the Hamming weight of x is with high probability concentrated around pd (e.g. by Hoeffding's inequality). Thus, in the first part of training the network will see, with high probability, only samples of the type $(x, \chi_S(x))$, and it will not see the second addend of $G_{S,T,\epsilon}$. When we change our input distribution to $\text{Rad}(1/2)^{\otimes d}$, the network will suddenly observe samples of the type $(x, \chi_T(x))$. Thus, the pre-training on p will not help determining the support of the new parity χ_T (in some sense the network will "forget" the first part of training). This intuition holds for all r-CL such that $p_1, ..., p_{r-1} < \epsilon$. We state our negative result for Hamming mixtures here informally, and refer to Section 5.4 for a formal statement and remarks.

Theorem 11 (Main negative result, informal)**.** *For each r-CL strategy with r bounded, there exists a Hamming mixture $G_{S,T,\epsilon}$ that is not learnable by any fully connected neural network of* $\text{poly}(d)$ *size and permutation-invariant initialization trained by the noisy gradient descent algorithm (see Def. 26) with* $\text{poly}(d)$ *gradient precision in* $\text{poly}(d)$ *steps.*

Inspired by the hardness of Hamming mixtures, we define another curriculum learning strategy, where, instead of having finitely many discrete curriculum steps, we gradually move the bias of the input distribution during training from a starting point p_0 to a final point p_T. We call this strategy a *continuous*-CL strategy.

Definition 24 (Continuous curriculum learning (C-CL))**.** *Let $p_0, p_T \in [0,1]$. We say that a neural network* $\text{NN}(x;\theta^t)$ *is trained by SGD with a C-CL(p_0, p_T, T) if θ^t follows the iterations in* (5.1) *with:*

$$\mathscr{D}^t = \text{Rad}\Big(p_0 + t \cdot \frac{p_T - p_0}{T}\Big), \qquad t \in [T]. \tag{5.4}$$

We conjecture that a well chosen C-CL might be beneficial for learning any Hamming mixture. A positive result for C-CL and comparison between r-CL and C-CL are left for future work.

5.3 Learning Parities

5.3.1 Theoretical Results

Our goal is to show that the curriculum strategy that we propose allows to learn k-parities with a computational complexity of $d^{O(1)}$. We prove two different results. In the first one, we consider SGD on the hinge loss and prove that a network with $\theta(d^2)$ hidden units can learn the k-parity problem in $d^{O(1)}$ computations, if trained with a well chosen 2-CL strategy. Let us state our first Theorem.

Theorem 12 (Hinge Loss)**.** *Let k, d be both even integers, such that $k \leq d/2$. Let* $\mathrm{NN}(x;\theta) = \sum_{i=1}^{N} a_i \sigma(w_i x + b_i)$ *be a 2-layers fully connected network with activation* $\sigma(y) := \mathrm{Ramp}(y)$ *(as defined in* (F.2)*) and* $N = \tilde{\theta}(d^2 \log(1/\delta))$[III]*. Consider training* $\mathrm{NN}(x;\theta)$ *with SGD on the hinge loss with batch size* $B = \tilde{\theta}(d^{10}/\epsilon^2 \log(1/\delta))$*. Then, there exists an initialization, a learning rate schedule, and a 2-CL strategy such that after* $T = \tilde{\theta}(d^6/\epsilon^2)$ *iterations, with probability* $1-3\delta$*, SGD outputs a network with generalization error at most* ϵ*.*

For our second Theorem, we consider another loss function, that is convenient for the analysis, namely the *covariance loss*, for which we give a definition here.

Definition 25 (Covariance loss)**.** *Let $f : \mathscr{X} \to \mathbb{R}$ be a target function and let $\hat{f} : \mathscr{X} \to \mathbb{R}$ be an estimator. Let*

$$\mathrm{cov}(f, \hat{f}, x, P_{\mathscr{X}}) := \Big(f(x) - \mathbb{E}_{x' \sim P_{\mathscr{X}}}[f(x')]\Big) \cdot \Big(\hat{f}(x) - \mathbb{E}_{x' \sim P_{\mathscr{X}}}[\hat{f}(x')]\Big),$$

where $P_{\mathscr{X}}$ is an input distribution supported in $\mathscr{X}$. We define the covariance loss as

$$L_{\mathrm{cov}}(f, \hat{f}, x, P_{\mathscr{X}}) := \max\{0, 1 - \mathrm{cov}(f, \hat{f}, x, P_{\mathscr{X}})\}.$$

We show that SGD on the covariance loss can learn the k-parity problem in $d^{O(1)}$ computations using a network with only $O(k)$ hidden units. The reduction of the size of the network, compared to the hinge loss case, allows to get a tighter bound on the computational cost (see Remark 12).

Theorem 13 (Covariance Loss)**.** *Let k, d be integers such that $k \leq d$ and k even. Let* $\mathrm{NN}(x;\theta) = \sum_{i=1}^{N} a_i \sigma(w_i x + b_i)$ *be a 2-layers fully connected network with activation* $\sigma(y) := \mathrm{ReLU}(y)$ *and* $N = \tilde{\theta}(k \log(1/\delta))$*. Consider training* $\mathrm{NN}(x;\theta)$ *with SGD on the covariance loss with batch size* $B = \tilde{\theta}(d^2 k^3/\epsilon^2 \log(1/\delta))$*. Then, there exists an initialization, a learning rate schedule, and a 2-CL strategy such that after* $T = \tilde{\theta}(k^4/\epsilon^2)$ *iterations, with probability* $1-3\delta$*, SGD outputs a network with generalization error at most* ϵ*.*

The proofs of Theorem 12 and Theorem 13 follow a similar outline. Firstly, we prove that training the first layer of the network for one step on one batch of size $d^{O(1)}$, sampled from a

[III] $\tilde{\theta}(d^c) = \theta(d^c \cdot \mathrm{poly}(\log(d)))$, for all $c \in \mathbb{R}$.

biased input distribution (with appropriate bias), allows to recover the support of the parity. We then show that training the second layer on the uniform distribution allows to achieve the desired generalization error under the uniform distribution. We remark that similar 'one-step' proofs appear e.g. in (Barak et al., 2022). We refer to Appendices F.1 and F.2 for restatements of the Theorems and their full proofs.

Remark 12. *Let us look at the computational complexity given by the two Theorems. Theorem 12 tells that we can learn k-parities in $dNB + (T-1)N = \tilde{\theta}(d^{19})$ computations. We remark that our result holds also for large k (we however need to assume k, d even and $k \leq d/2$, for technical reasons). On the other hand, Theorem 13 tells that we can learn k-parities in $\tilde{\theta}(d^3 k^8)$, which is lower than the bound given by Theorem 12. Furthermore, the proof holds for all $k \leq d$. The price for getting this tighter bound is the use of a loss that (to the best of our knowledge) is not common in the machine learning literature, and that is particularly convenient for our analysis.*

Remark 13. *We remark that our proofs extend to the gradient descent model with bounded gradient precision, used in (Abbe and Sandon, 2020a), with gradient precision bounded by $d^{O(1)}$ (see Remark 20 in Appendix F.1).*

Remark 14. *Let us comment on the p_1 (i.e. the bias of the initial distribution) that we used. In both Theorems we take p_1 close to 1. In Theorem 12 we take $p_1 \approx 1-\theta(1/d)$, and the proof is constructed specifically for this value of p_1. In Theorem 13, the proof holds for any $p_1 \in (1/2, 1)$ and the asymptotic complexity in d does not depend on the specific choice of p_1. However, to get* $\mathrm{poly}(k)$ *complexity we need to take $p_1 = 1-\theta(1/k)$, while we get* $\exp(k)$ *complexity for all $p_1 = \theta_{d,k}(1)$.*

Our theoretical analysis captures a fairly restricted setting: in our proofs we use initializations and learning schedules that are convenient for the analysis. We conduct experiments to verify the usefulness of our CL strategy in more standard settings of fully connected architectures.

5.3.2 Empirical Results

In all our experiments we use fully connected ReLU networks and we train them by SGD on the square loss.

In Figure 5.1, we compare different curriculum strategies for learning 20-parities over 100 bits, with a fixed architecture, i.e. a 2-layer ReLU network with 100 hidden units. We run a 2-steps curriculum strategy for 3 values of p_1, namely $p_1 = 39/40, 19/20, 1/20$. In all the 2-CL experiments we train on the biased distribution until convergence, and then we move to the uniform distribution. We observe that training with an initial bias of $p_1 = 39/40$ allows to learn the 20-parity in $16,000$ epochs. One can see that during the first part of training (on the biased distribution), the test error under the uniform distribution stays at $1/2$ (orange line), and then drops quickly to zero when we start training on the uniform distribution. This trend of hidden progress followed by a sharp drop has been already observed in the context

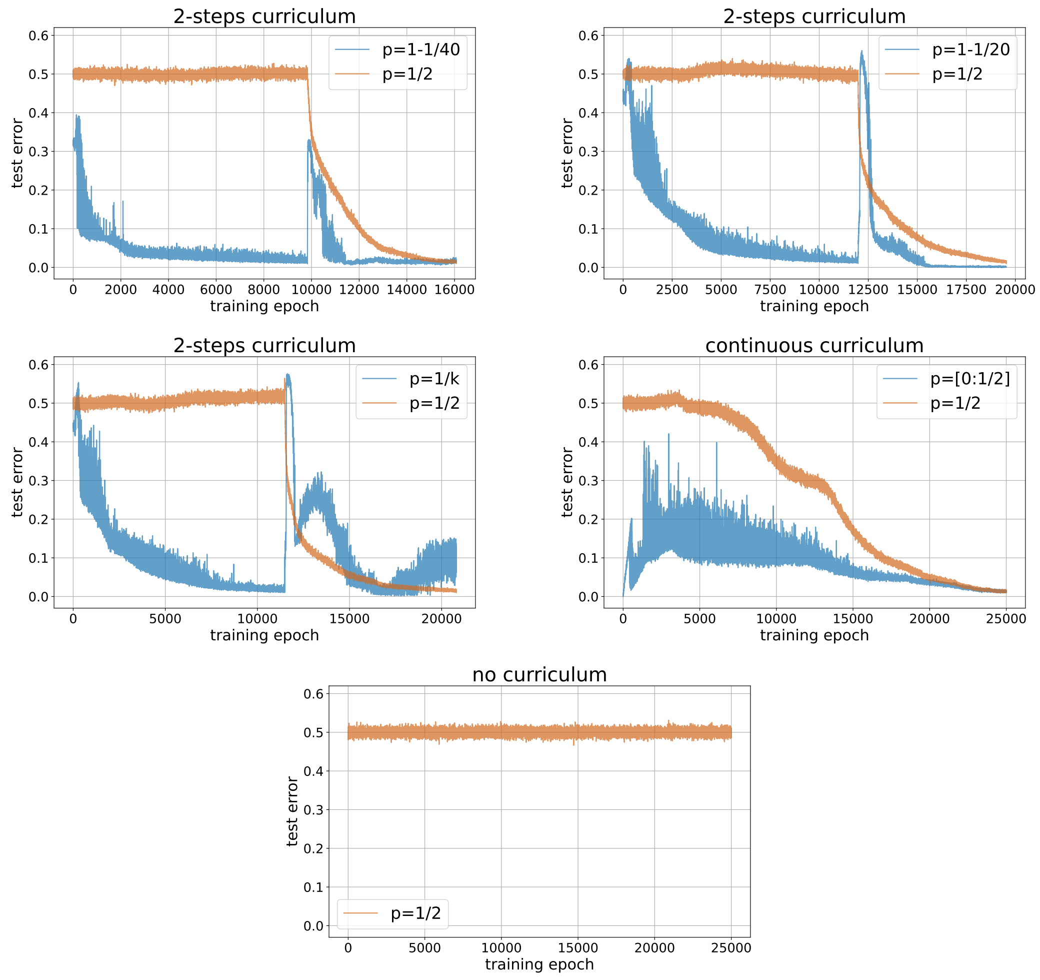

Figure 5.1: Learning 20-parities with 2-steps curriculum, with initial bias $p_1 = 39/40$ (top-left), $p_1 = 19/20$ (top-center), $p_1 = 1/20$ (top-right), with continuous curriculum (bottom-left) and with no curriculum (bottom-right). In all plots, we use a 2-layers ReLU MLP with batch size 1024, input dimension 100, and 100 hidden units.

of learning parities with SGD in the standard setting with no-curriculum (Barak et al., 2022). Here, the length of the 'hidden progress' phase is controlled by the length of the first phase of training. Interestingly, when training with continuous curriculum, we do not have such hidden progress and the test error under the uniform distribution decreases slowly to zero. With no curriculum, the network does not achieve non-trivial correlation with the target in $25,000$ epochs. We refer to the Appendix of (Cornacchia and Mossel, 2023) for further experiments with smaller batch size and 3-layers networks.

In Figure 5.2 we study the convergence time of a 2-CL strategy on a 2-layers ReLU network for different values of the input dimension (d) and size of the parity (k). We take two slightly different settings. In the plot on the left, we take a fixed initial bias $p_1 = 1/16$ and $N = 2^k$ hidden units. On the right we take $p_1 = 1 - \frac{1}{2k}$ initial bias and an architecture with $N = d$

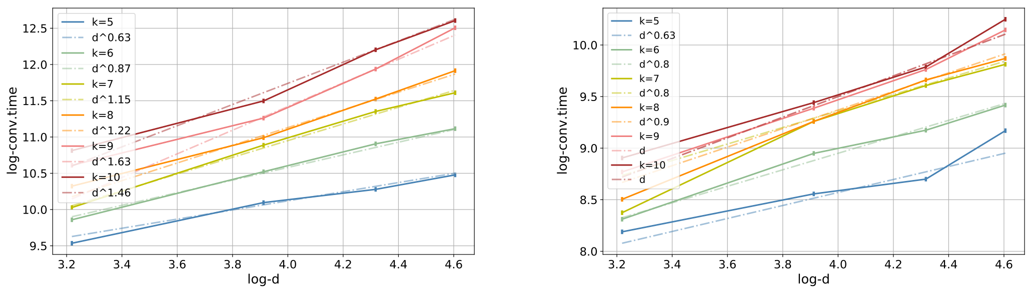

Figure 5.2: Convergence time for different values of d, k. Left: we take $p_1 = 1/16$ and a 2-layers ReLU architecture with with $N = 2^k$ hidden units. Right: we take $p_1 = 1 - \frac{1}{2k}$ and a 2-layers ReLU architecture with $N = d$ hidden units.

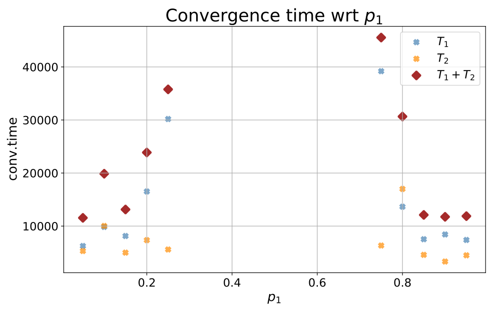

Figure 5.3: Convergence time with respect to the initial bias p_1. We compute the convergence time for learning a 10-parity over 100 bits with a 2-layer ReLU network. We omitted all points with convergence time above 100,000.

hidden units. The convergence time is computed as $T_1 + T_2$, where T_1 and T_2 are the number of steps needed to achieve training error below 0.01 in the first and second part of training, respectively. We compute the convergence time for $k = 5,6,7,8,9,10$ and $d = 25,50,75,100$, and for each k we plot the convergence time with respect to d in log-log scale. Each point is obtained by averaging over 10 runs. We observe that for each k, the convergence time scales (roughly) polynomially as d^{c_k}, with c_k varying mildly with k.

In Figure 5.3, we study the convergence time of a 2-CL strategy for different values of the initial bias p_1. We consider the problem of learning a 10-parity over 100 bits with a 2-layers ReLU network with $N = 100$ hidden units. As before, we computed the convergence time as $T_1 + T_2$, where T_1 and T_2 are the number of steps needed to achieve training error below 0.01 in the first and second part of training, respectively. We ran experiments for $p_1 = 0.001, 0.05, 0.1, 0.15, ..., 0.95, 0.999$. We omitted from the plot any point for which the convergence time exceeded 100,000 iterations: these correspond to p_1 near 1/2 and $p_1 = 0.001, 0.999$. Each point is obtained by averaging over 10 runs. We observe that the convergence time is smaller for p_1 close to 0 or to 1. Moreover, T_2 has modest variations across different p_1's.

5.4 Learning Hamming Mixtures

In this section we consider the class of functions defined in Def. 23 and named Hamming mixtures. We consider a specific descent algorithm, namely the noisy GD algorithm with batches (used also in (Abbe, Kamath, et al., 2021; Abbe and Sandon, 2020a)). We give a formal definition here of noisy GD with curriculum.

Definition 26 (Noisy GD with CL)**.** *Consider a neural network* $\mathrm{NN}(.;\theta)$*, with initialization of the weights* θ^0*. Given an almost surely differentiable loss function, the updates of the noisy GD algorithm with learning rate* γ_t *and gradient range* A *are defined by*

$$\theta^{t+1} = \theta^t - \gamma_t \left(\mathbb{E}_{x^t}[\nabla_{\theta^t} L(\theta^t, f, x^t)]_A + Z^t \right), \tag{5.5}$$

where for all $t \in \{0, ..., T-1\}$*,* Z^t *are i.i.d.* $\mathcal{N}(0,\tau^2)$*, for some* τ*, and they are independent from other variables,* $x^t \sim \mathcal{D}^t$*, for some time-dependent input distribution* $\mathcal{D}^t$*,* f *is the target function, from which the labels are generated, and by* $[.]_A$ *we mean that whenever the argument is exceeding* A *(resp.* $-A$*) it is rounded to* A *(resp.* $-A$*). We call* A/τ *the gradient precision. In the noisy-GD algorithm with* r*-CL, we choose* $\mathcal{D}^t$ *according to Def. 21.*

Let us state our hardness result for learning Hamming mixtures with r-CL strategies with r bounded.

Theorem 14. *Assume the network observes samples generated by* $G_{S,V,\epsilon}(x)$ *(see Def. 23), where* $|S| = k_S$*,* $|V| = k_V$ *such that* $k_S, k_V = o(\sqrt{d})$*, and* $|S \cap V| = 0$*. Then, for any* r*-CL*$(\bar{T}, \bar{p})$ *with* r *bounded and* $p_r = 1/2$*, there exists an* ϵ *such that the noisy GD algorithm with* r*-CL*$(\bar{T}, \bar{p})$ *(as in* (5.5)*) on a fully connected neural network with* P *weights and permutation-invariant initialization, after* T *training steps, outputs a network* $\mathrm{NN}(x, \theta^T)$ *such that*

$$\left| \mathbb{E}_{x \sim \mathrm{Rad}(1/2)^{\otimes d}} \left[G_{S,V,\epsilon}(x) \cdot \mathrm{NN}(x;\theta^T) \right] \right| \le \frac{2AT\sqrt{P}}{\tau} \left(\binom{d}{k_V}^{-1/2} + e^{-d\delta^2} \right) + \frac{2k_S k_V}{d} + O(d^{-2}),$$

where A, τ *are the gradient range and the noise level in the noisy-GD algorithm and* δ *is a constant.*

The proof uses an SQ-like lower bound argument for noisy GD, in a similar flavour of (Abbe and Boix-Adserà, 2022; Abbe, Cornacchia, et al., 2022). We refer to Appendix F.3 for the full proof.

Remark 15. *In Theorem 14, the neural network can have any fully connected architecture and any activation such that the gradients are well defined almost everywhere. The initialization can be from any distribution that is invariant to permutations of the input neurons.*

For the purposes of $\mathbb{E}\left[G_{S,V,\epsilon}(x) \cdot \mathrm{NN}(x;\theta^T)\right]$, it is assumed that the neural network outputs a guess in $\{\pm 1\}$. This can be done with any form of thresholding, e.g. taking the sign of the value of the output neuron.

Remark 16. *One can remove the* $\frac{2k_S k_V}{d}$ *term in the right hand side by further assuming e.g. that set* S *is supported on the first* $d/2$ *coordinates and set* V *on the last* $d/2$ *coordinates. This also allows to weaken the assumption on the cardinality of* S *and* V*. We formalize this in the following Corollary.*

Corollary 6. *Assume the network observes samples generated by* $G_{S,V,\epsilon}(x)$*, where* $S \subseteq \{1,...,d/2\}$*, and* $V \subseteq \{d/2+1,...,d\}$ *(where we assumed* d *to be even for simplicity). Denote* $k_V = |V|$*. Then, for any* r*-CL*$(\bar{T},\bar{p})$ *with* r *bounded and* $p_r = 1/2$*, there exists an* ϵ *such that the noisy GD algorithm with* r*-CL*$(\bar{T},\bar{p})$ *(as in* (5.5)*) on a fully connected neural network with* P *weights and permutation-invariant initialization, after* T *training steps, outputs a network* $\mathrm{NN}(x,\theta^T)$ *such that*

$$\left|\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[G_{S,V,\epsilon}(x)\cdot \mathrm{NN}(x;\theta^T)\right]\right| \le \frac{2AT\sqrt{P}}{\tau}\left(\binom{d/2}{k_V}^{-1/2}+e^{-d\delta^2}\right),$$

for some $\delta>0$.

The proof of Corollary 6 is deferred to Appendix F.4.

Theorem 14 and Corollary 6 state failure at the weakest form of learning, i.e. achieving correlation better than guessing in the asymptotic of large d. More specifically, it tells that if the network size, the number of training steps and the gradient precision (i.e. A/τ) are such that $\frac{AT\sqrt{P}}{\tau} = o(d^{-k_V/2})$, then the network achieves correlation with the target of $o_d(1)$ under the uniform distribution. Corollary 7 follows immediately from the Theorem.

Corollary 7. *Under the assumptions of Theorem 14, if* $k_V = \omega_d(1)$ *(i.e.* k_V *grows with* d*),* $P, A/\tau, T$ *are all polynomially bounded in* d*, then*

$$\left|\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[G_{S,V,\epsilon}(x)\cdot \mathrm{NN}(x;\theta^T)\right]\right| = o_d(1), \tag{5.6}$$

i.e. in $\mathrm{poly}(d)$ *computations the network will fail at weak-learning* $G_{S,V,\epsilon}$*.*

We conjecture that if we take instead a C-CL strategy with an unbounded number of curriculum steps, we can learn efficiently (i.e. in poly(d) time) any $G_{S,V,\epsilon}$ (even with $k_V = \omega_d(1)$ and for any ϵ). Furthermore, we believe this conjecture to hold for any bounded mixture, i.e. any function of the type:

$$\sum_{m=1}^{M} \chi_{S_m}(x)\mathbb{1}(\epsilon_{m-1}d \le H(x) < \epsilon_m d), \tag{5.7}$$

with $S_1,...,S_M$ being distinct sets of coordinates, $0=\epsilon_0<\epsilon_1...<\epsilon_M\le 1$, and M bounded.

5.5 Conclusion and Future Work

In this work, we mainly focused on learning parities and Hamming mixtures with r-CL strategies with bounded r. Some natural questions arise, for instance: does the depth of the network help? What it the optimal number of curriculum steps for learning parities? We leave to future work the analysis of C-CL with unboundedly many curriculum steps and the comparison between r-CL and C-CL. In the previous Section, we also raised a conjecture concerning the specific case of Hamming mixtures.

Furthermore, we believe that our results can be extended to more general families of functions. First, consider the set of k-Juntas, i.e., the set of functions that depend on k out of d coordinates. This set of functions contains the set of k-parities so it is at least as hard to learn. Moreover, as in the case of parities, Juntas are correlated with each of their inputs for generic p, see e.g. (Mossel et al., 2004). So it is natural to expect that curriculum learning can learn such functions in time $d^{O(1)}2^{O(k)}$ (the second term is needed since there is a doubly exponential number of Juntas on k bits). We further believe that our general idea of introducing correlation among subsets of the input coordinates to facilitate learning may apply to more general non-Boolean settings.

In this work we propose to learn parities using a mixture of product distributions, but there are other ways to correlate samples that may be of interest. For example, some works in PAC learning showed that, even for the uniform measure, samples that are generated by a random walk often lead to better learning algorithms (Arpe and Mossel, 2008; Bshouty et al., 2005). Do such random walk based algorithms provide better convergence for gradient based methods?

An important limitation of the curriculum strategy presented in this Chapter is that it requires an oracle that provides labeled samples from arbitrary product measures. However, in applications one usually has a fixed dataset and would like to select samples in a suitable order, to facilitate learning. In a work in preparation (Abbe, Cornacchia, and Lotfi, 2023), we consider learning parities on inputs generated from the following *mixed* distribution:

$$\mathscr{D} = \lambda \operatorname{Rad}(p)^{\otimes d} + (1-\lambda)\operatorname{Rad}(1/2)^{\otimes d}, \tag{5.8}$$

with $\lambda \in (0,1)$ and p close to 1. We consider a curriculum strategy that consists of restricting the training set to samples with large Hamming weight (i.e.those coming from the $\operatorname{Rad}(p)^{\otimes d}$ part of the distribution) for the first part of training, and subsequently training on the whole dataset. We compare this strategy to the standard setting with no-curriculum, where we train on the whole dataset at every step. We show that for small λ ($\lambda = O(d^{-2})$) a similar proof strategy as the one of this Chapter allows to prove separation between the two settings (with and without curriculum).

6 Conclusion

In this thesis, we have explored two different learning problems, namely statistical inference on graph models and gradient descent on neural networks. Despite their differences, these problems share a common goal of analyzing the measures that characterize the fundamental limits of learning.

In the first part of the thesis, we have investigated the problem of spin synchronization on graphs and how the erasure side-information affects the correlations between spins at distant sites. We have showed that the boundary irrelevance (BI) property does not hold for every tree, while a weaker property, the boundary irrelevance in the non-reconstruction regime (BIN), holds for every tree. We conjectured that the (BIN) holds for all graphs. Furthermore, we have used the (BI) to characterize the limiting entropy of the sparse SBM with two symmetric communities. As mentioned in the conclusion section of Chapter 2, future research directions include 1) closing Conjecture 1, 2) investigating the (BI) property for a wider range of graphs beyond regular and Poisson trees, as well as exploring the possibility of characterizing the set of graphs for which the (BI) holds, 3) examining more general settings with non-binary labels, and determining whether the (BI) property holds for large alphabets in all regimes, 4) exploring more general definitions of side-information that reveal interactions of sets of node variables, and analyzing the role of observation symmetries in these settings.

While the investigation of spin synchronization on graphs and the study of the erasure side-information are interesting, it would be valuable to consider more realistic models that incorporate additional complexities, such as time-varying graphs or dynamic interactions between nodes (Kim et al., 2014). These models could help us gain a deeper understanding of the complex behavior of synchronization and correlation on graphs, and may have important implications for applications in fields such as neuroscience and social network analysis. Moreover, while it is essential to understand the information-theoretic limits of learning, it is also important to take into account the computational complexity of the learning process. Developing efficient algorithms for learning on graphs is particularly important in this context, given the large-scale and complex nature of many real-world networks.

In the second part of the thesis, we have studied the problem of learning Boolean target functions with gradient descent on fully connected neural networks. We have introduced the notion of Initial Alignment (INAL) and shown that noisy gradient descent cannot learn a target function in polynomial time if there is no noticeable INAL between the network and the target. We have shown that the result can be extended beyond Boolean inputs for finite depth networks trained with the correlation loss. For this latter extension, we considered a slightly different measure than the INAL, namely the initial gradient alignment.

The Initial Alignment and the Initial Gradient Alignment can be easily estimated by Monte-Carlo simulations, which makes them suitable for applications. It would be interesting to analyse how these or related measures (e.g. alignment after few steps of training) can predict the performance of an architecture on a given dataset before training. Indeed, having a measure that can provide insights into the training dynamics of a neural network before actually training it could save a lot of time and computational resources. Furthermore, such measures could aid in the design of new neural architectures (as it is done in Neural Architecture Search (NAS) (Elsken et al., 2019)), allowing for more efficient and effective models to be created.

Additionally, we investigated the generalization error under the canonical holdout (a specific case of distribution shift) and propose the "low-degree implicit bias hypothesis" for certain architectures, such as MLPs and Transformers. In particular, in the case of PVR under canonical holdout (Figure 4.2), we observed that the performance of Transformers is similar to the performance of MLPs. In other words, Transformers do not seem to leverage their self-attention mechanisms to compare effectively different parts of the input strings. However, a thorough investigation of the power of self-attention mechanisms in this and in more general settings is still missing. It would be thus interesting to find and analyze a task where a Transformer architecture outperform a simpler architecture (e.g. MLP) under distribution shift.

Appendices Part

A Appendix on Block Models

A.1 Background on Information Theory

Let us recall the definitions of entropy and mutual information of discrete random variables.

Definition 27 (Entropy)**.** *The entropy of a discrete random variable U, that takes value in the alphabet $\mathcal{U}$, is defined as:*

$$H(U) := \sum_{u\in\mathcal{U}} p(u) \log \frac{1}{p(u)}, \tag{A.1}$$

where $p(u) = \mathbb{P}(U = u)$.

The entropy of a random variable U is the average level of uncertainty associated to the possible outcomes of U.

Definition 28 (Conditional Entropy)**.** *Suppose U, V are discrete random variables taking values in $\mathcal{U}$ and $\mathcal{V}$, respectively, with joint distribution $p(u, v) = \mathbb{P}(U = u, V = v)$ and conditional distribution $p(u|v) = \mathbb{P}(U = u|V = v)$. The conditional entropy of U given V is defined as*

$$H(U|V) := \sum_{u\in\mathcal{U}, v\in\mathcal{V}} p(u, v) \log \frac{1}{p(u|v)}. \tag{A.2}$$

Definition 29 (Mutual Information)**.** *The mutual information between random variables U and V is defined as*

$$I(U;V) := H(U) - H(U|V), \tag{A.3}$$

where $H(U)$ and $H(U|V)$ are the entropy of U and the conditional entropy of $U|V$, respectively.

The mutual information measures the amount of information that can be obtained about one random variable by observing another one.

A.2 Proof of Proposition 1

Let us consider $\sigma_0 = X_0$, σ_t be the leaves at depth t generated according to the broadcasting model described in the Introduction and σ^ϵ be the side-information. We prove the equivalent statement: $I(\sigma_0;\sigma_t|\sigma^\epsilon) \le I(\sigma_0;\sigma_t)$ for every $\epsilon \in [0,1]$. Let us condition on one realization of the survey σ_S and consider: $I(\sigma_0;\sigma_t|\sigma^\epsilon = \sigma_S)$. Note that $I(\sigma_0;\sigma_t|\sigma^\epsilon = \sigma_S) = I(\sigma_0;\sigma_{t\setminus S}|\sigma^\epsilon = \sigma_S)$, where by $t \setminus S$ we mean the set of leaves at depth t with no ancestor in S. For brevity, denote $U = t \setminus S$.
Following the "stringification" idea of (Evans et al., 2000) (Theorem 1.3), we build a tree $\hat{T}$ such that all the leaves in U are 'separated' from all the nodes in S, but the original tree structure for the vertices S and the tree structure of U are maintained. To do this, one can color red all descendants of all nodes in S, and color blue all leaves in U. Then, starting from the uncolored nodes at depth $t-1$, we color red the nodes that have all red children, and blue the nodes with all blue children. If a node has mixed offspring, we split it into two nodes and we attach the blue descendants to the first one and the red descendants to the other, and we color the two nodes consequently. We proceed this way up to the root of the tree, that we leave uncolored.
By Theorem 1.3 in (Evans et al., 2000) (DPI) we have $I_T(\sigma_0;\sigma_S,\sigma_U) \le I_{\hat{T}}(\sigma_0;\sigma_S,\sigma_U)$, and by construction, $I_T(\sigma_0;\sigma_U) = I_{\hat{T}}(\sigma_0;\sigma_U)$ and $I_T(\sigma_0;\sigma_S) = I_{\hat{T}}(\sigma_0;\sigma_S)$. Moreover, on $\hat{T}$ σ_S is independent of σ_U given σ_0, thus $I_{\hat{T}}(\sigma_0;\sigma_S,\sigma_U) \le I_{\hat{T}}(\sigma_0;\sigma_U) + I_{\hat{T}}(\sigma_0;\sigma_S)$. Putting these together and applying chain rule gives the result.

A.3 Stringy Tree Computations

In this Section, we use the following notation:

- L_t: leaves at depth t;
- R: root;
- S_t: survey at depth $\le t$;
- N: number of surveyed branches;
- T_i^t: depth of first surveyed node in branch i, conditioned on branch i being surveyed before depth t.

Notice that

$$N \sim \mathrm{Bin}(1-\epsilon^t, d^t), \tag{A.4}$$

$$\mathbb{P}(T_i^t = k) = \frac{1-\epsilon}{1-\epsilon^t}\epsilon^{k-1}, \qquad k \in [t]. \tag{A.5}$$

Also, let X_t be the label of a node at depth t, then

$$\mathbb{P}(X_t=+|R=+)=\frac{1}{2}+\frac{1}{2}(1-2\eta)^t; \tag{A.6}$$
$$\mathbb{P}(X_t=-|R=+)=\frac{1}{2}-\frac{1}{2}(1-2\eta)^t. \tag{A.7}$$

We will show that iff $d\epsilon(1-2\eta)^2<1$

$$\lim_{t\to\infty}\mathbb{E}^+_{S_t}\left[\frac{\mathbb{P}(R=+|S_t)}{\mathbb{P}(R=-|S_t)}\right]=\lim_{t\to\infty}\mathbb{E}^+_{S_t,L_t}\left[\frac{\mathbb{P}(R=+|S_t,L_t)}{\mathbb{P}(R=-|S_t,L_t)}\right] \tag{A.8}$$

where $\mathbb{E}^+$ denotes the expectation conditioned on the true root label being $+$.

A.3.1 Survey Only

Notice that $\frac{\mathbb{P}(R=+|S_t)}{\mathbb{P}(R=-|S_t)}=\frac{\mathbb{P}(S_t|R=+)}{\mathbb{P}(S_t|R=-)}$.

$$\mathbb{E}^+_{S_t}\left[\frac{\mathbb{P}(S_t|R=+)}{\mathbb{P}(S_t|R=-)}\right]=\mathbb{E}^+_{X_S,N,T_1,\dots,T_N}\left[\frac{\mathbb{P}(S_t|R=+)}{\mathbb{P}(S_t|R=-)}\right] \tag{A.9}$$
$$=\mathbb{E}_N\left[\prod_{i\in[N]}\mathbb{E}^+_{X_{T_i},T_i}\left[\frac{\mathbb{P}(X_{T_i}|R=+)}{\mathbb{P}(X_{T_i}|R=-)}\right]\right] \tag{A.10}$$
$$=\mathbb{E}_N\left[\prod_{i\in[N]}\mathbb{E}^+_{X_{T_i},T_i}\left[\frac{\frac{1}{2}+\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}{\frac{1}{2}-\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}\right]\right] \tag{A.11}$$
$$=\mathbb{E}_N\left[E_S^N\right] \tag{A.12}$$
$$=(\epsilon^t+(1-\epsilon)^tE_S)^{d^t}. \tag{A.13}$$

where we denoted $E_S:=\mathbb{E}^+_{X_{T_i},T_i}\left[\frac{\frac{1}{2}+\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}{\frac{1}{2}-\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}\right]$ and where we used the moment generating function of binomial to deduce the last equality.

$$\mathbb{E}^+_{X_{T_i}}\left[\frac{\frac{1}{2}+\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}{\frac{1}{2}-\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}\right]=\frac{\left(\frac{1}{2}+\frac{1}{2}(1-2\eta)^{T_i}\right)^2}{\frac{1}{2}-\frac{1}{2}(1-2\eta)^{T_i}}+\frac{\left(\frac{1}{2}-\frac{1}{2}(1-2\eta)^{T_i}\right)^2}{\frac{1}{2}+\frac{1}{2}(1-2\eta)^{T_i}} \tag{A.14}$$
$$=1+4\frac{(1-2\eta)^{2T_i}}{1-(1-2\eta)^{2T_i}}. \tag{A.15}$$

Thus,

$$E_S=\mathbb{E}_{T_i}\left[1+4\frac{(1-2\eta)^{2T_i}}{1-(1-2\eta)^{2T_i}}\right] \tag{A.16}$$
$$=1+4\frac{1-\epsilon}{1-\epsilon^t}\sum_{k=1}^{t}\frac{(1-2\eta)^{2k}}{1-(1-2\eta)^{2k}}\epsilon^{k-1}. \tag{A.17}$$

Notice that $\frac{(1-2\eta)^2}{1-(1-2\eta)^2} \leq \sum_{k=1}^{t} \frac{(1-2\eta)^{2k}}{1-(1-2\eta)^{2k}} \epsilon^{k-1} \leq \frac{(1-2\eta)^2}{(1-(1-2\eta)^2)} \frac{1}{(1-(1-2\eta)^2\epsilon)}$, thus $E_S = \theta(1)$.

A.3.2 Survey and Leaves

Again, we have that $\frac{\mathbb{P}(R=+|S_t,L_t)}{\mathbb{P}(R=-|S_t,L_t)} = \frac{\mathbb{P}(S_t,L_t|R=+)}{\mathbb{P}(S_t,L_t|R=-)}$, thus

$$\mathbb{E}^+_{S_t,L_t}\left[\frac{\mathbb{P}(S_t,L_t|R=+)}{\mathbb{P}(S_t,L_t|R=-)}\right] = \mathbb{E}_N\left[\prod_{i\in[N]} \mathbb{E}^+_{X_{T_i},T_i}\left[\frac{\frac{1}{2}+\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}{\frac{1}{2}-\frac{1}{2}X_{T_i}(1-2\eta)^{T_i}}\right]\prod_{j=N}^{d^t}\mathbb{E}^+_{X_t}\left[\frac{\frac{1}{2}+\frac{1}{2}X_t(1-2\eta)^t}{\frac{1}{2}-\frac{1}{2}X_t(1-2\eta)^t}\right]\right] \tag{A.18}$$

$$= \mathbb{E}_N\left[E_S^N E_L^{d^t-N}\right] = E_L^{d^t}\mathbb{E}_N\left[\left(\frac{E_S}{E_L}\right)^N\right] \tag{A.19}$$

$$= (\epsilon^t E_L + (1-\epsilon^t)E_S)^{d^t}, \tag{A.20}$$

where we denoted $E_L := \mathbb{E}^+_{X_t}\left[\frac{\frac{1}{2}+\frac{1}{2}X_t(1-2\eta)^t}{\frac{1}{2}-\frac{1}{2}X_t(1-2\eta)^t}\right]$ and where again we used the moment generating function of binomial to find the last equality.
Note that from previous calculations

$$E_L = 1 + 4\frac{(1-2\eta)^{2t}}{1-(1-2\eta)^{2t}} \tag{A.21}$$

A.3.3 Conclusion

Let us look at the ratio

$$\frac{\mathbb{E}^+_{S_t,L_t}\left[\frac{\mathbb{P}(R=+|S_t,L_t)}{\mathbb{P}(R=-|S_t,L_t)}\right]}{\mathbb{E}^+_{S_t}\left[\frac{\mathbb{P}(R=+|S_t)}{\mathbb{P}(R=-|S_t)}\right]} = \left(\frac{\epsilon^t E_L + (1-\epsilon^t)E_S}{\epsilon^t + (1-\epsilon)^t E_S}\right)^{d^t} \tag{A.22}$$

$$= \left(1 + \frac{4\epsilon^t(1-2\eta)^{2t}}{(\epsilon^t + (1-\epsilon)^t E_S)(1-(1-2\eta)^{2t})}\right)^{d^t} \tag{A.23}$$

$$\sim \left(1 + C_t\epsilon^t(1-2\eta)^{2t}\right)^{d^t} \tag{A.24}$$

$$\sim e^{C_t\left(d\epsilon(1-2\eta)^2\right)^t}, \tag{A.25}$$

where $C_t = \theta(1)$ (in particular $C_t := \frac{4}{E_S}$).
We observe that $e^{C_t\left(d\epsilon(1-2\eta)^2\right)^t} \to 1$ iff $d\epsilon(1-2\eta)^2 < 1$.

A.4 Proof of Theorem 1

Let us denote $f(\epsilon) = H(X|G,\omega^\epsilon)$, where similarly as before ω^ϵ is a BEC$_\epsilon$-survey that reveals the true label of each node independently with probability $1-\epsilon$. Note that $f(1) = H(X|G)$.

Let us replace the single parameter ϵ by a set of parameters $\vec{\epsilon} = (\epsilon_u)_{u \in V(G)}$ (for each vertex u, X_u is revealed with probability $1-\epsilon_u$), and let us denote $\omega^{\epsilon}_{\sim u} = \{\omega^{\epsilon}_v : v \in V(G), v \neq u\}$ and $X_{\sim u} = \{X_v : v \in V(G), v \neq u\}$. Then

$$f(\vec{\epsilon}) = (1-\epsilon_u) H(X|G, X_u, \omega^{\epsilon}_{\sim u}) + \epsilon_u H(X|G, \omega^{\epsilon}_{\sim u}) \tag{A.26}$$

and by chain rule

$$\begin{aligned} \frac{\partial}{\partial \epsilon_u} f(\vec{\epsilon}) &= H(X|G, \omega^{\epsilon}_{\sim u}) - H(X|G, X_u, \omega^{\epsilon}_{\sim u}) && \text{(A.27)} \\ &= H(X_u, X_{\sim u}|G, \omega^{\epsilon}_{\sim u}) - H(X_{\sim u}|G, X_u, \omega^{\epsilon}_{\sim u}) && \text{(A.28)} \\ &= H(X_u|G, \omega^{\epsilon}_{\sim u}). && \text{(A.29)} \end{aligned}$$

Then, setting $\epsilon_u = \epsilon$ for all $u \in V(G)$, we get by symmetry

$$f'(\epsilon) = \sum_{u \in V(G)} H(X_u|G, \omega^{\epsilon}_{\sim u}) = nH(X_1|G, \omega^{\epsilon}_{\sim 1}). \tag{A.30}$$

Thus, by bounded convergence

$$\lim_{n \to \infty} \frac{1}{n} H(X|G) = \int_0^1 \lim_{n \to \infty} H(X_1|G, \omega^{\epsilon}_{\sim 1}) d\epsilon. \tag{A.31}$$

Take $k = \frac{\log n}{10 \log 2(a+b)}$ small enough compared to n, such that the neighborhood of vertex 1 at depth k is a tree with high probability (this is for instance proved as Proposition 2 in (Mossel et al., 2015)), and denote such neighborhood by T_k. Specifically, w.h.p. T_k is a Galton-Watson tree with Poisson$\left(\frac{a+b}{2}\right)$ offspring distribution, rooted at 1, and the labels in X_{T_k} are distributed as BOT with flip probability $\frac{b}{a+b}$. Moreover, let X_{L_k} be the vertices at distance exactly k from 1, and let $\omega^{\epsilon}_{\sim 1, T_k}$ denote the survey on nodes at distance at most k from 1 (excluding 1). We bound the integrand by the following:

$$H(X_1|T_k, \omega^{\epsilon}_{\sim 1, T_k}, X_{L_k}) + o_k(1) \le H(X_1|G, \omega^{\epsilon}_{\sim 1}) \le H(X_1|T_k, \omega^{\epsilon}_{\sim 1, T_k}). \tag{A.32}$$

For the inequality on the right, we simply removed conditioning terms and thus increased the conditional entropy, specifically we ignored any information from the graph or from the survey on nodes at distance $\ge k$ to 1. The inequality on the left requires the following lemma, that is a direct consequence of Proposition 2 and Lemma 4.7 in (Mossel et al., 2015).

Lemma 3. $H(X_1|G, \omega^{\epsilon}_{\sim 1}, X_{L_k}) = H(X_1|T_k, \omega^{\epsilon}_{\sim 1, T_k}, X_{L_k}) + o_k(1)$.

In words, Lemma 3 states that after conditioning on the leaves, the information coming from the graph outside T_k (including non-edges) becomes negligible, i.e. the model is asymptotically a Markov field. By Theorem 2, if $\frac{(a-b)^2}{2(a+b)} \le 1$ or $\frac{(a-b)^2}{2(a+b)} \ge \alpha^*$, then (BI) holds for $(T_k, 1, \frac{a-b}{a+b}, \text{BEC}_\epsilon)$, for all $\epsilon < 1$, thus the leftmost and the rightmost terms in (A.32) are asymptotically equal. This means that the limit in the integrand in (A.31) exists for all $\epsilon \in (0,1)$, thus (i) holds.

B Appendix on Initial Alignment

B.1 Proof of Proposition 3

B.1.1 Outline of the Proof

The main goal of the proof is to estimate the dominant term (as d approaches infinity) of $\mathrm{INAL}(\chi_k, \sigma)$, and show that it is indeed noticeable, for any fixed k. We initially use Jensen's inequality to lower bound the INAL with the following

$$\mathrm{INAL}(\chi_k, \sigma) \geq \mathbb{E}\Big[\mathbb{E}_{|\theta|,x}\big[\chi_k(x)\sigma(w^T x + b) \mid \mathrm{sgn}(\theta)\big]^2\Big], \tag{B.1}$$

where for brevity we denoted $\theta = (w, b)$, $|\theta|$ and $\mathrm{sgn}(\theta)$ are $(d+1)$-dimensional vectors such that $|\theta|_i = |\theta_i|$ and $\mathrm{sgn}(\theta)_i = \mathrm{sgn}(\theta_i)$, for all $i \leq d+1$. By denoting $|w|_{>k}, x_{>k}$ the coordinates of $|w|$ and x respectively that do not appear in χ_k, and by $G := \sum_{i=1}^k w_i x_i + b$ we observe that

$$\mathbb{E}_{|w|_{>k}, x_{>k}}[\sigma(w^T x + b)] = \mathbb{E}_{Y \sim \mathcal{N}(0, 1-\frac{k}{d})}[\sigma(G + Y)], \tag{B.2}$$

since $\sum_{i=k+1}^d w_i x_i$ is indeed distributed as $\mathcal{N}(0, 1 - \frac{k}{d})$. We call the RHS the "$d$-Gaussian smoothing" of σ and we denote it by $\Sigma_d(z) := \mathbb{E}_{Y \sim \mathcal{N}(0,1-\frac{k}{d})}[\sigma(z+Y)]$. We will compare it to the "ideal" Gaussian smoothing denoted by $\Sigma(z) := \mathbb{E}_{Y \sim \mathcal{N}(0,1)}[\sigma(z+Y)]$.

For polynomially bounded σ, we can prove that Σ_d has some nice properties (see Lemma 4), specifically it is C^∞ and polynomially bounded and it uniformly converges to Σ as $d \to \infty$. These properties crucially allow to write Σ_d in terms of its Taylor expansion around 0, and bound the coefficients of the series for large d. In fact, we show that there exists a constant $P > k$, such that if we split the Taylor series of Σ_d at P as

$$\Sigma_d(G) = \sum_{\nu=0}^{P} a_{\nu,d} G^\nu + R_{P,d}(G), \tag{B.3}$$

(where $a_{\nu,d}$ are the Taylor coefficients and $R_{P,d}$ is the remainder in Lagrange form), and take the expectation over $|\theta|_{\leq k}$ as:

$$\mathbb{E}_{|\theta|_{\leq k}, x_{\leq k}}\left[\chi_k(x)\Sigma_d(G)\right] = \sum_{\nu=0}^{P} a_{\nu,d}\mathbb{E}_{|\theta|_{\leq k}, x_{\leq k}}\left[\chi_k(x)G^{\nu}\right] + \mathbb{E}_{|\theta|_{\leq k}, x_{\leq k}}\left[R_{P,d}(G)\right] \tag{B.4}$$

$$=: A + B, \tag{B.5}$$

then A is $\Omega(d^{-P/2})$ (Proposition 11), and B is $O(d^{-P/2-1/2})$ (Proposition 12), uniformly for all values of $\operatorname{sgn}(\theta)$. For A we use the observation that $\mathbb{E}_{|\theta|_{\leq k}, x_{\leq k}}\left[\chi_k(x)G^{\nu}\right] = 0$ for all $\nu < k$ (Lemma 6), and the fact that $|a_{P,d}| > 0$ for d large enough (due to hypothesis b in Definition 6 and the continuity of Σ_d in the limit of $d \to \infty$, given by Lemma 4). For B, we combine the concentration of Gaussian moments and the polynomial boundness of all derivatives of Σ_d. Taking the square of (B.5) and going back to (B.1), one can immediately conclude that $\mathrm{INAL}(\chi_k, \sigma)$ is indeed noticeable.

B.1.2 Useful Lemmas

For an activation $\sigma : \mathbb{R} \to \mathbb{R}$, we denote its v-Gaussian smoothing as

$$\Sigma_v(t) := \mathbb{E}_{Y \sim \mathcal{N}(0,v)}[\sigma(Y+t)]. \tag{B.6}$$

We also write $\Sigma := \Sigma_1$ for brevity. We work with functions that are *polynomially bounded*, ie., such that there exists a polynomial P with $|\sigma(x)| < P(x)$ holding for all $x \in \mathbb{R}$. We will use the fact that such polynomial can be assumed wlog to be of the form $|\sigma(x)| < Cx^{\ell} + C$ for some $C > 0$ and $\ell \in \mathbb{N}_{\geq 0}$ (since any polynomial can be upper bounded by a polynomial of such form). Note that if σ is a measurable, polynomially bounded function, then Σ_v is well defined for every $v > 0$.

We now state the intermediate step in the proof of Proposition 3:

Lemma 4 (Conditions on Σ and Σ_v)**.** *If σ is a measurable, polynomially bounded function, then it satisfies the following conditions:*

i) $\Sigma_v \in C^{\infty}(\mathbb{R})$ *for every* $v > 0$;

ii) For every $k \in \mathbb{N}_{\geq 0}$ *and* $v > 0$, $\Sigma_v^{(k)}(t) := \frac{\partial^k}{\partial t^k}\Sigma_v(t)$ *is polynomially bounded. Furthermore, this bound is uniform, that is,* $|\Sigma_v^{(k)}(t)| < Ct^{\ell} + C$ *holds for every* $t \in \mathbb{R}$ *and every* $1/2 \leq v \leq 1$, *for some* C, ℓ *that do not depend on* v.

iii) For all $k \in \mathbb{N}_{\geq 0}$, *it holds* $|\Sigma_{1-\epsilon}^{(k)}(0) - \Sigma^{(k)}(0)| = O(\epsilon)$.

Lemma 4 is then used in the proof of the following Lemma.

Lemma 5. *Let σ be expressive (according to Definition 6). Then, for every $k \geq 0$ and $P \geq k$ such that $\Sigma^{(P)}(0) \neq 0$, it holds that* $\mathrm{INAL}(\chi_k, \sigma) = \Omega(d^{-P})$.

In particular, from Lemma 5 it follows that if σ is expressive, then it is correlating. Furthermore, since by condition b) in Definition 6 for every k we have $\Sigma^{(k)}(0) \neq 0$ or $\Sigma^{(k+1)}(0) \neq 0$, by Lemma 5 it holds $\mathrm{INAL}(\chi_k, \sigma) = \Omega(d^{-(k+1)})$ and σ is 1-strongly correlating.

In the following subsections we prove Lemma 4 and Lemma 5.

B.1.3 Proof of Lemma 4

In the following let ϕ_v denote the density function of $\mathcal{N}(0, v)$, ie., $\phi_v(t) = \frac{1}{\sqrt{2v\pi}} \exp\left(-\frac{t^2}{2v}\right)$. Note the relation to the standard Gaussian density $\phi = \phi_1$ where $\phi_v(t) = \frac{1}{\sqrt{v}}\phi(t/\sqrt{v})$.

We recall some useful facts about the derivatives of ϕ_v. First, it is well known that for ϕ it holds $\phi^{(k)}(t) = P_k(t)\phi(t)$ for some polynomial P_k of degree k. This formula extends to ϕ_v according to

$$\phi_v^{(k)}(t) = \frac{1}{\sqrt{v}} \frac{\mathrm{d}^k}{\mathrm{d}t^k} \phi(t/\sqrt{v}) = v^{-k/2-1/2}\phi^{(k)}(t/\sqrt{v}) = v^{-k/2-1/2} P_k(t/\sqrt{v})\phi(t/\sqrt{v}) \tag{B.7}$$

$$= v^{-k/2} P_k(t/\sqrt{v})\phi_v(t)\,. \tag{B.8}$$

i) Let us write $\phi_v(t) = (\phi_{v/2} * \phi_{v/2})(t)$ where $*$ denotes the convolution in $\mathbb{R}$, i.e. $(g*h)(y) = \int_{\mathbb{R}} g(x)h(y-x)dx$. Thus,

$$\Sigma_v = \sigma * \phi_v = (\sigma * \phi_{v/2}) * \phi_{v/2}. \tag{B.9}$$

Now, $\sigma * \phi_{v/2}$ is in $L_1(\mathbb{R})$, since σ is measurable and polynomially bounded. Furthermore, $\phi_{v/2}$ is in $L_1(\mathbb{R})$ and $C^\infty(\mathbb{R})$. Therefore, by formulas for derivatives of convolution, $\Sigma_v \in C^\infty(\mathbb{R})$.

ii) Let us start with the claim that $\Sigma_v^{(k)}$ is polynomially bounded for every v and k. For that, we recall some facts. First, it is easy to establish by direct computation that if σ is polynomially bounded, then $\Sigma_v = \sigma * \phi_v$ is also polynomially bounded. Furthermore, if P is any polynomial, then also $\sigma * (P\phi_v)$ is polynomially bounded (this can be seen, eg., by observing that for every P and every $v' > v$ there exists C such that $|P\phi_v| \leq C\phi_{v'}$).

Accordingly, using (B.7) and (B.9) we have that

$$\Sigma_v^{(k)} = (\sigma * \phi_{v/2}) * \phi_{v/2}^{(k)} = (\sigma * \phi_{v/2}) * (P_{k,v}\phi_{v/2}) \tag{B.10}$$

is polynomially bounded.

Let us move to the second claim with uniform bound. For that let $k \geq 0$ and $1/2 \leq v \leq 1$. Let $v' := v - 1/4$ and note that $1/4 \leq v' \leq 3/4$. Then, we have the sequence of bounds on functions which hold pointwise:

$$|\Sigma_v^{(k)}| = \left|(\sigma * \phi_{1/4}) * \phi_{v'}^{(k)}\right| \le C_1 \left(|\sigma * \phi_{1/4}| * |P_k(x/\sqrt{v'})|\phi_{v'}\right) \tag{B.11}$$

$$\le C_1 \left(|\sigma * \phi_{1/4}| * (C_2 + C_2(x/\sqrt{v})^{2\ell})\phi_{v'}\right) \tag{B.12}$$

$$\le C_3 \left(|\sigma * \phi_{1/4}| * (C_4 + C_4 x^{2\ell})\phi\right), \tag{B.13}$$

which is now bounded by a polynomial which does not depend on v.

iii) Recall,

$$\Sigma_v^{(k)}(0) = \int_{-\infty}^{\infty} (\phi_{v/2} * \sigma)(x) \cdot \frac{\partial^k}{\partial t^k} \phi_{v/2}(x+t)\Big|_{t=0} dx, \tag{B.14}$$

where we denoted by $\phi_{v/2}^{(k)}$ the k-th derivative of $\phi_{v/2}$. Firstly, note that

$$\frac{\partial^k}{\partial t^k} \phi_{v/2}(x+t)\Big|_{t=0} = \frac{\partial^k}{\partial (x+t)^k} \phi_{v/2}(x+t)\Big|_{t=0} = \phi_{v/2}^{(k)}(x). \tag{B.15}$$

Let us give a formula for the k-th derivative of the Gaussian density:

$$\phi_v^{(k)}(x) = \phi_v(x) \cdot (-1)^k v^{-2k} \cdot \sum_{l=0}^{k} D_{l,k} \left(\frac{x}{\sqrt{v}}\right)^{k-l}, \tag{B.16}$$

where $D_{l,k}$ is a constant that does not depend on v, specifically

$$D_{l,k} := B_{(2k+l)\frac{1-(-1)^l}{2}} \cdot 2^{\frac{l}{2}} \cdot \frac{\Gamma(\frac{l+1}{2})}{\Gamma(\frac{1}{2})} \cdot \cos\left(\frac{l\pi}{2}\right) \tag{B.17}$$

where $\Gamma(.)$ denotes the Gamma function and B_n are the Bernoulli numbers. The exact values of the $D_{l,k}$ will not be relevant for this proof. Thus,

$$\Sigma_v^{(k)}(0) := \int_{-\infty}^{\infty} (\phi_{v/2} * \sigma)(x) P_{v/2,k}(x) \phi_{v/2}(x) dx, \tag{B.18}$$

where we denoted $P_{v/2,k}(x) = (-1)^k v^{-2k} \cdot \sum_{l=0}^{k} D_{l,k} \left(\frac{x}{\sqrt{v}}\right)^{k-l}$. On the other hand,

$$\Sigma_1^{(k)}(0) := \int_{-\infty}^{\infty} (\phi_{v/2} * \sigma)(x) P_{1-v/2,k}(x) \phi_{1-v/2}(x) dx, \tag{B.19}$$

and

$$|\Sigma_v^{(k)}(0) - \Sigma_1^{(k)}(0)| = \left| \int_{-\infty}^{\infty} (\phi_{v/2} * \sigma)(x) \cdot \left(P_{v/2,k}(x)\phi_{v/2}(x) - P_{1-v/2,k}(x)\phi_{1-v/2}(x)\right) \right|. \tag{B.20}$$

We note that

$$
\begin{aligned}
P_{1-v/2,k}(x) =& \frac{(1-\frac{v}{2})^{-2k}}{(\frac{v}{2})^{-2k}}\left(\frac{v}{2}\right)^{-2k}(-1)^k \qquad \text{(B.21)}\\
&\cdot\left(\sum_{l=0}^{k} D_{l,k}\left(\frac{x}{\sqrt{v/2}}\right)^{k-l} + D_{l,k}\left[\left(\frac{x}{\sqrt{1-v/2}}\right)^{k-l} - \left(\frac{x}{\sqrt{v/2}}\right)^{k-l}\right]\right) \qquad \text{(B.22)}\\
=& \frac{(1-\frac{v}{2})^{-2k}}{(\frac{v}{2})^{-2k}} P_{v/2,k}(x) \qquad \text{(B.23)}\\
&+\left(1-\frac{v}{2}\right)^{-2k}(-1)^k\sum_{l=0}^{k} D_{l,k}\left(\left(\frac{x}{\sqrt{1-v/2}}\right)^{k-l} - \left(\frac{x}{\sqrt{v/2}}\right)^{k-l}\right). \qquad \text{(B.24)}
\end{aligned}
$$

Recalling $\epsilon = 1-v$, and expanding for such ϵ we get

$$
\begin{aligned}
&\left(1+\frac{2\epsilon}{1-\epsilon}\right)^{-2k} P_{v/2,k}(x) + (1+\epsilon)^{-2k}\frac{(-1)^k}{2^{-2k}}\sum_{l=0}^{k} D_{l,k}\, x^{k-l}\frac{(1-\epsilon)^{k-l}-(1+\epsilon)^{k-l}}{(1+\epsilon)^{\frac{k-l}{2}}(1-\epsilon)^{\frac{k-l}{2}}} \qquad \text{(B.25)}\\
=&\left(1-4k\frac{\epsilon}{1-\epsilon}+o(\epsilon)\right)P_{v/2,k}(x) \qquad \text{(B.26)}\\
&+(1-2k\epsilon+o(\epsilon))\frac{(-1)^k}{2^{-2k}}\sum_{l=0}^{k} D_{l,k}\, x^{k-l}\frac{-2(k-l)\epsilon+(\epsilon)}{(1+\frac{k-l}{2}\epsilon+o(\epsilon))(1-\frac{k-l}{2}\epsilon+o(\epsilon))} \qquad \text{(B.27)}\\
=&\left(1-4k\frac{\epsilon}{1-\epsilon}\right)P_{v/2,k}(x)+O(\epsilon)\mathscr{P}_k(x), \qquad \text{(B.28)}
\end{aligned}
$$

where $\mathscr{P}_k(x)$ is a polynomial in x of degree $\leq k$. Moreover,

$$
\begin{aligned}
\phi_{1-v/2}(x) =& \frac{e^{-\frac{x^2}{v}}}{\sqrt{2\pi v/2}}\cdot\sqrt{\frac{v/2}{1-v/2}}\cdot e^{-\frac{x^2}{2}\left(\frac{1}{1-\frac{v}{2}}-\frac{2}{v}\right)} \qquad \text{(B.29)}\\
=&\ \phi_{v/2}(x)\cdot\left(1-\frac{2\epsilon}{1+\epsilon}\right)^{1/2}\cdot e^{x^2\frac{2\epsilon}{(1+\epsilon)(1-\epsilon)}} \qquad \text{(B.30)}\\
=&\ \phi_{v/2}(x)\cdot\left(1-\frac{\epsilon}{1+\epsilon}+o(\epsilon)\right)\cdot\left(1+x^2\frac{2\epsilon}{(1+\epsilon)(1-\epsilon)}+o(\epsilon)x^4\right) \qquad \text{(B.31)}\\
=&\ \phi_{v/2}(x)\cdot\left(1+(x^2-1)O(\epsilon)\right). \qquad \text{(B.32)}
\end{aligned}
$$

Plugging these bounds in the previous expression, we get

$$|\Sigma_v^{(k)}(0)-\Sigma_1^{(k)}(0)| \tag{B.33}$$
$$=\left|\int_{-\infty}^{\infty}(\phi_{v/2}*\sigma)(x)\cdot\left(P_{v/2,k}(x)\phi_{v/2}(x)-P_{1-v/2,k}(x)\phi_{v/2}(x)\left(1+(x^2-1)O(\epsilon)\right)\right)\right| \tag{B.34}$$
$$=\left|\int_{-\infty}^{\infty}(\phi_{v/2}*\sigma)(x)\phi_{v/2}(x)\cdot\left(P_{v/2,k}(x)-P_{1-v/2,k}(x)\left(1+(x^2-1)O(\epsilon)\right)\right)\right| \tag{B.35}$$
$$=\left|\int_{-\infty}^{\infty}(\phi_{v/2}*\sigma)(x)P_{v/2,k}(x)\phi_{v/2}(x)\cdot\left(1-(1-O(\epsilon)+(x^2-1)O(\epsilon))+O(\epsilon)\mathscr{P}_k(x)\right)\right| \tag{B.36}$$
$$=O(\epsilon)\,. \tag{B.37}$$

□

B.1.4 Proof of Lemma 5

Note that we only need to show that $\mathrm{INAL}(\chi_k,\sigma)=\Omega(d^{-P})$ for the first index P such that $P\geq k$ and $\Sigma^{(P)}(0)\neq 0$. By Definition 6, we only need with two cases $P=k$ and $P=k+1$. From now on, let us consider a fixed pair of k and P.

We denote by $x\in\{\pm1\}^d$ the vector of all inputs, by $w\in\mathbb{R}^d$ the vector of all weights and by $b\in\mathbb{R}$ the bias. Additionally, we denote $\tau_i:=\mathrm{sgn}(w_i)$, and by $\tau\in\{\pm1\}^d$ the vector of all weight signs. Recall that we consider $w_i,b\overset{iid}{\sim}\mathcal{N}(0,\frac{1}{d})$ and that for $g,h:\{\pm1\}^d\to\{\pm1\}$ and $\mathcal{U}^d$ being the uniform distribution over the hypercube, we denote $\langle g,h\rangle=\mathbb{E}_{x\sim\mathcal{U}^d}[g(x)h(x)]$. We have

$$\mathrm{INAL}(\chi_k,\sigma)=\mathbb{E}_{w,b}\left[\langle\chi_k,\sigma\rangle^2\right] \tag{B.38}$$
$$=\mathbb{E}_{|w|,\tau,|b|,\mathrm{sgn}(b)}\left[\langle\chi_k,\sigma\rangle^2\right] \tag{B.39}$$
$$\overset{(C.S.)}{\geq}\mathbb{E}_{\tau,\mathrm{sgn}(b)}\left[\mathbb{E}_{|w|,|b|}\left[\langle\chi_k,\sigma\rangle\mid\tau,\mathrm{sgn}(b)\right]^2\right], \tag{B.40}$$

where (B.40) follows by Cauchy-Schwartz inequality. We will prove a lower bound on the inner expectation $\left(\mathbb{E}_{|w|,|b|}\langle\chi_k,\sigma\rangle\right)^2$ which is independent of τ and $\mathrm{sgn}(b)$. Accordingly, from now on consider τ and $\mathrm{sgn}(b)$ to be fixed at arbitrary values.

Let $T:=\{1,\ldots,k\}$ and denote by x_T the coordinates of x contained in T, and by $x_{\sim T}:=x_{T^C}$ the coordinates of x that are not contained in T and hence do not appear in the monomial χ_T. Similarly, we denote by $|w|_T,|w|_{\sim T}$ the coordinates of $|w|$ that appear (respectively do not appear) in set T. We proceed,

$$\mathbb{E}_{|w|,|b|}\langle \chi_T, \sigma\rangle = \mathbb{E}_{x,|w|,|b|}\left[\chi_T(x)\cdot \sigma\left(\sum_{i\in[d]} w_i x_i + b\right)\right] \tag{B.41}$$

$$= \mathbb{E}_{|w|_T, x_T, |b|}\left[\chi_T(x)\cdot \mathbb{E}_{|w|_{\sim T}, x_{\sim T}}\sigma\left(\sum_{i\in[d]} w_i x_i + b\right)\right] \tag{B.42}$$

Observe that $\sum_{i\notin T} w_i x_i \sim \mathcal{N}(0, \frac{d-k}{d})$, and denote $\Sigma_d(z) := \Sigma_{1-\frac{k}{d}}(z) = \mathbb{E}_{Y\sim\mathcal{N}(0,\frac{d-k}{d})}[\sigma(z+Y)]$. Moreover, let $G := \sum_{i\in T} w_i x_i + b$. Then,

$$\mathbb{E}_{|w|,|b|}\langle \chi_T, \sigma\rangle = \mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)\Sigma_d(G)\right]. \tag{B.43}$$

Since, by condition i) in Lemma 4, function Σ_d is C^∞ and therefore C^P, we apply Taylor's theorem with Lagrange remainder and write

$$\Sigma_d(z) = \sum_{\nu=0}^{P} a_{\nu,d} z^\nu + R_{P,d}(z), \tag{B.44}$$

where $a_{\nu,d} = \frac{\Sigma_d^{(\nu)}(0)}{\nu!}$ and

$$R_{P,d}(z) = \frac{\Sigma_d^{(P+1)}(\xi_z)}{(P+1)!} z^{P+1} \qquad \text{for some } |\xi_z| \le |z|. \tag{B.45}$$

Plugging this in (B.43), we get

$$\mathbb{E}_{|w|,|b|}\langle \chi_T, \sigma\rangle = \sum_{\nu=0}^{P} a_{\nu,d}\mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)G^\nu\right] + \mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)R_{P,d}(G)\right]. \tag{B.46}$$

The following two propositions give the asymptotic characterization of the first and second term in (B.46).

Proposition 11.

$$\sum_{\nu=0}^{P} a_{\nu,d}\mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)G^\nu\right] = C(P)(-1)^{C'(\tau_T,\mathrm{sgn}(b))} d^{-P/2} + O(d^{-P/2-1/2}). \tag{B.47}$$

where $C(P) \neq 0$ and $C'(\tau_T, \mathrm{sgn}(b)) \in \mathbb{Z}$ are constants that do not depend on d.

Proposition 12.

$$\mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)R_{P,d}(G)\right] = O(d^{-P/2-1/2}). \tag{B.48}$$

Before proving Propositions 11 and 12, let us see how Lemma 5 follows from them. But this is clear: substituting into (B.46), we have

$$\left(\mathbb{E}_{|w|,|b|}\langle \chi_T, \sigma\rangle\right)^2 = C(P)^2 d^{-P} + O(d^{-P-1/2}) = \Omega(d^{-P}), \tag{B.49}$$

where the claimed bound does not depend on τ nor on $\mathrm{sgn}(b)$.

Proof of Proposition 11

The main step for proving Proposition 11 is the computation of $\langle \chi_T, G^\nu\rangle$, for $\nu \le P$. This is summarized in the following formula.

Lemma 6. *We have:*

$$\mathbb{E}_{|w|_T,|b|,x_T} \chi_T(x) G^\nu = \begin{cases} 0 & \text{if } \nu < k \\ C(\nu)(-1)^{C'(\tau_T,\mathrm{sgn}(b))} d^{-\nu/2} & \text{if } \nu \ge k\,, \end{cases} \tag{B.50}$$

where $C(\nu) > 0$.

Let us first see how to finish the proof once Lemma 6 is established. Recall that $a_{\nu,d} = \frac{\Sigma^{(\nu)}_{1-k/d}(0)}{\nu!}$ and let $a_\nu := \frac{\Sigma^{(\nu)}(0)}{\nu!}$. We are considering a sum with $P+1$ terms, so let $s_\nu := a_{\nu,d}\mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)G^\nu\right]$. Accordingly, our objective is to show that

$$\sum_{\nu=0}^{P} s_\nu = C(P)(-1)^{C'(\tau_T,\mathrm{sgn}(b))} d^{-P/2} + O(d^{-P/2-1/2}). \tag{B.51}$$

We do that by considering the terms s_ν one by one. For $\nu < k$, from (B.50) we immediately have $s_\nu = 0$.

For $k \le \nu < P$, by Definition 6 recall that the only possible case is $P = k+1$ and $\Sigma^{(k)}(0) = 0$. Then, applying condition iii) from Lemma 4,

$$|a_{\nu,d}| = \left|\frac{\Sigma^{(\nu)}_{1-k/d}(0) - \Sigma^{(\nu)}(0)}{\nu!}\right| = O(d^{-1}), \tag{B.52}$$

which together with (B.50) gives $|s_\nu| = O(d^{-P/2-1/2})$.

Finally, for $\nu = P$, by assumption we have $a_P \ne 0$. Then, by condition iii), we have $|a_{P,d} - a_P| = O(1/d)$ and (B.50) gives us the correct form for s_P and the whole expression.

All that is left is the proof of Lemma 6.

Proof of Lemma 6. The proof proceeds by using the linearity of expectation and independence and expanding the formula for G^{ν}. Recall that we assumed wlog that $T = \{1, \dots, k\}$ and let $z_i := w_i x_i$ for $i \le k$ and $z_{k+1} := b$:

$$\mathbb{E}_{|w|_T,|b|,x_T} \chi_T(x) G^{\nu} = \mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} x_i\right)\left(\sum_{i=1}^{k} w_i x_i + b\right)^{\nu} \tag{B.53}$$

$$= \sum_{I=(i_1,\dots,i_\nu)\in[k+1]^\nu} \mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} x_i\right)\left(\prod_{i\in I} z_i\right). \tag{B.54}$$

Let us focus on a single term of the sum in (C.66) for $I = (i_1, \dots, i_\nu) \in [k+1]^\nu$. For $j = 1, \dots, k+1$, let $\alpha_j = \alpha_j(I) := |\{m : i_m = j\}|$. Accordingly, we can rewrite a term from (C.66) as

$$\mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} x_i\right)\left(\prod_{i\in I} z_i\right) \tag{B.55}$$

$$= \mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} w_i^{\alpha_i} x_i^{\alpha_i+1}\right) b^{\alpha_{k+1}} \tag{B.56}$$

$$= \left(\prod_{i=1}^{k} \tau_i^{\alpha_i}\right) \operatorname{sgn}(b)^{\alpha_{k+1}} \left(\prod_{i=1}^{k} \mathbb{E}_{|w|_i}\left[|w|_i^{\alpha_i}\right] \cdot \mathbb{E}_{x_i}\left[x_i^{\alpha_i+1}\right]\right) \mathbb{E}_{|b|}\left[|b|^{\alpha_{k+1}}\right]. \tag{B.57}$$

Since $\mathbb{E}[x_i^{\alpha_i+1}] = 0$ if α_i is even, for a term in (C.68) to be non-zero it is necessary that α_i is odd for every $1 \le i \le k$. Consequently, since $\sum_{i=1}^{k+1} \alpha_i = \nu$, in any non-zero term the parity of α_{k+1} is equal to the parity of $\nu - k$. Therefore, every non-zero term is of the form

$$\mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} x_i\right)\left(\prod_{i\in I} z_i\right) = \left(\prod_{i=1}^{k} \tau_i\right) \operatorname{sgn}(b)^{\mathbb{1}[\nu-k \text{ odd}]} \cdot \left(\prod_{i=1}^{k} \mathbb{E}_{|w|_i}\left[|w_i|^{\alpha_i}\right]\right) \mathbb{E}_{|b|}\left[|b|^{\alpha_{k+1}}\right] \tag{B.58}$$

$$= (-1)^{C'(\tau_T, \operatorname{sgn}(b))} \cdot \left(\prod_{i=1}^{k} \mathbb{E}_{|w|_i}\left[|w_i|^{\alpha_i}\right]\right) \mathbb{E}_{|b|}\left[|b|^{\alpha_{k+1}}\right]. \tag{B.59}$$

We now establish the first case from (B.50). If $\nu < k$, then since $\nu = \sum_{i=1}^{k+1} \alpha_i$ at least one of α_i, $1 \le i \le k$ must be zero, and therefore even. Consequently, each term in (C.66) is zero and it follows that $\mathbb{E}_{|w|_T,|b|,x_T} \chi_T(x) G^{\nu} = 0$.

On the other hand, for $\nu \ge k$, there exists a non-zero term, for example taking $\alpha_1 = \ldots = \alpha_k = 1$ and $\alpha_{k+1} = \nu - k$. Take any such term arising from $I \in [k+1]^\nu$. Since $w_i, b \sim \mathcal{N}(0, 1/d)$, we have $\mathbb{E}_{|w|_i}\left[|w_i|^j\right], E_{|b|}\left[|b|^j\right] = C_j \cdot d^{-j/2}$ for some $C_j > 0$ for every fixed j. Substituting in (B.59) and using $\nu = \sum_{i=1}^{k+1} \alpha_i$, we get

$$\mathbb{E}_{|w|_T,|b|,x_T} \left(\prod_{i=1}^{k} x_i\right)\left(\prod_{i\in I} z_i\right) = (-1)^{C'(\tau_T, \operatorname{sgn}(b))} C_I d^{-\nu/2} \tag{B.60}$$

for some $C_I > 0$. Therefore, $C(\nu)(-1)^{C'(\tau_T, \operatorname{sgn}(b))} d^{-\nu/2}$ with $C(\nu) > 0$ follows since it is a sum of at most $(k+1)^{\nu}$ positive terms. □

Proof of Proposition 12

Let D be a positive constant. We apply the decomposition

$$\left|\mathbb{E}_{|w|_T,|b|,x_T}\left[\chi_T(x)R_{P,d}(G)\right]\right| \leq \mathbb{E}_{|w|_T,|b|,x_T}\left[\left|R_{P,d}(G)\right|\cdot \mathbb{1}(|G|\leq D)\right] \tag{B.61}$$

$$+\mathbb{E}_{|w|_T,|b|,x_T}\left[\left|R_{P,d}(G)\right|\cdot \mathbb{1}(|G|> D)\right] \tag{B.62}$$

The proposition follows from Lemmas 7 and 8 applied to an arbitrary value of D, eg., $D=1$.

Lemma 7. *For any $D>0$,*

$$\mathbb{E}_{|w|_T,|b|,x_T}\left[\left|R_{P,d}(G)\right|\cdot \mathbb{1}(|G|\leq D)\right] = O\left(d^{-\frac{P+1}{2}}\right). \tag{B.63}$$

Proof. Let us observe that for a fixed b, $G\sim\mathcal{N}(b,\frac{k}{d})$, thus

$$\mathbb{E}_{|w|_T,x_T}\left[|R_{P,d}(G)|\mathbb{1}(|G|\leq D)\right] = \mathbb{E}_{y\sim\mathcal{N}(b,\frac{k}{d})}\left[|R_{P,d}(y)|\mathbb{1}(|y|\leq D)\right]. \tag{B.64}$$

Recall that $R_{P,d}(x)=\frac{\Sigma_d^{(P+1)}(\xi_x)}{(P+1)!}x^{P+1}$ for some $|\xi_x|\leq|x|$. Thus,

$$\mathbb{E}_{y\sim\mathcal{N}(b,\frac{k}{d})}\left[|R_{P,d}(y)|\mathbb{1}(|y|\leq D)\right] \leq \sup_{|y|\leq D}\frac{|\Sigma_d^{(P+1)}(y)|}{(P+1)!}\cdot \mathbb{E}_{y\sim\mathcal{N}(b,\frac{k}{d})}|y|^{P+1}. \tag{B.65}$$

On the one hand, assuming that $d\geq 2k$, we have $\Sigma_d=\Sigma_\nu$ for some $1/2\leq\nu\leq1$, and thus using the common polynomial bound in property ii) $\sup_{|y|\leq D}|\Sigma_d^{(P+1)}(y)|\leq M_D$, where the constant M_D does not depend on d. On the other hand,

$$\mathbb{E}_{y\sim\mathcal{N}(b,\frac{k}{d})}|y|^{P+1} = d^{-\frac{P+1}{2}}\cdot\mathbb{E}_y\left|\sqrt{d}\cdot y\right|^{P+1} \tag{B.66}$$

$$\leq d^{-\frac{P+1}{2}}\cdot 2^{P+1}\cdot\left(|\sqrt{d}b|^{P+1}+\mathbb{E}_{z\sim\mathcal{N}(0,k)}|z|^{P+1}\right) \tag{B.67}$$

$$= d^{-\frac{P+1}{2}}\cdot 2^{P+1}\cdot\left(|\sqrt{d}b|^{P+1}+\frac{(2k)^{\frac{P+1}{2}}\Gamma(\frac{P+2}{2})}{\sqrt{\pi}}\right), \tag{B.68}$$

where in the last equation we plugged the (P+1)-th central moment of the Gaussian distribution (see, eg., Winkelbauer, 2012). Since $|\sqrt{d}b|$ is also distributed like an absolute value of $\mathcal{N}(0,1)$, taking the expectation over $|b|$, we get that for fixed P,k,

$$\mathbb{E}_{|w|_T,|b|,x_T}\left[\left|R_{P,d}(G)\right|\cdot \mathbb{1}(|G|\leq D)\right] = O\left(d^{-\frac{P+1}{2}}\right). \tag{B.69}$$

□

Lemma 8. *For any constant $D > 0$, there exist $C_1, C_2 > 0$ such that*

$$\mathbb{E}_{|w|_T,|b|,x_T}\Big[\big|R_{P,d}(G)\big| \cdot \mathbb{1}(|G| > D)\Big] \le C_1 \exp(-C_2 d). \tag{B.70}$$

Proof. By Cauchy-Schwartz inequality,

$$\mathbb{E}_{|w|_T,|b|,x_T}\left[|R_{P,d}(G)|\mathbb{1}(|G| > D)\right] \overset{(C.S.)}{\le} \mathbb{E}_{|w|_T,|b|,x_T}[R_{P,d}(G)^2]^{1/2} \cdot \Pr_{|w|_T,|b|,x_T}[|G| > D]^{1/2}\,. \tag{B.71}$$

For the first term, we use the universal polynomial bound from property ii):

$$\begin{aligned}
\left|\mathbb{E}_{|w|_T,|b|,x_T}[R_{P,d}(G)^2]\right| &= \mathbb{E}_{|w|_T,|b|,x_T}\left[\left(\frac{\sup_{|y|\le|G|}\Sigma_d^{(P+1)}(y)}{(P+1)!}|G|^{P+1}\right)^2\right] &\text{(B.72)}\\
&\le \mathbb{E}_{|w|_T,|b|,x_T}\left[\left(\frac{\sup_{|y|\le|G|} C y^{2\ell}+C}{(P+1)!}|G|^{P+1}\right)^2\right] &\text{(B.73)}\\
&= \mathbb{E}_{|w|_T,|b|,x_T}\left[\left(\frac{C G^{2\ell}+C}{(P+1)!}|G|^{P+1}\right)^2\right] = O_d(1)\,, &\text{(B.74)}
\end{aligned}$$

using a similar reasoning as in Lemma 7.

On the other hand, writing $G = G' + |b|$, we have

$$\begin{aligned}
\Pr_{|w|_T,|b|,x_T}[|G| > D] &\le \Pr_{|b|}[|b| > D/2] + \Pr_{|w|_T,x_T}[|G'| > D/2] &\text{(B.75)}\\
&\le 2\Pr_{y\sim\mathcal{N}(0,1/d)}[|y| > D/2] \le 4\exp(-D^2 d/8)\,. &\text{(B.76)}
\end{aligned}$$

We get desired bound putting together (B.72) and (B.76). □

B.2 Expressivity of Common Activation Functions

In this section we show that ReLU and sign are expressive. It is clear that both of these functions are polynomially bounded, so we only need to analyze their Hermite expansions for condition b) in Definition 6. In both cases we do it by writing a closed form for $\Sigma^{(k)}(0)$.

Proposition 13. $\mathrm{ReLU}(x) := \max\{0, x\}$ *is expressive.*

Proof. We will see that in the case $\sigma = \mathrm{ReLU}$ we have $\Sigma(z) = \frac{z}{2} + \frac{z}{2}\,\mathrm{erf}(z) + \frac{1}{2\sqrt{\pi}}\exp(-z^2)$.

Indeed,

$$\Sigma(z) = \int_{-\infty}^{\infty} \mathbb{1}(z+y \geq 0)(z+y)\phi(y)\,\mathrm{d}y = \int_{-z}^{\infty} (z+y)\phi(y)\,\mathrm{d}y = z\Phi(z) + \phi(z) \tag{B.77}$$

$$= \frac{z}{2} + \frac{z\,\mathrm{erf}(z/\sqrt{2})}{2} + \phi(z)\,. \tag{B.78}$$

Using well-known Taylor expansions of erf and ϕ, this results in

$$\Sigma^{(k)}(0) = \begin{cases} \frac{1}{2} & \text{if } k=1\,, \\ \frac{(-1)^{k/2+1}}{\sqrt{2\pi}2^{k/2}(k-1)(k/2)!} & \text{if } k \text{ is even,} \\ 0 & \text{otherwise.} \end{cases} \tag{B.79}$$

In particular, $\Sigma^{(k)}(0) \neq 0$ for every even k and ReLU is expressive. □

Proposition 14. *The sign function* sgn(x) *is expressive.*

Proof. In this case, similarly, we have

$$\Sigma(z) = -\int_{-\infty}^{-z} \phi(z)\,\mathrm{d}z + \int_{-z}^{\infty} \phi(z)\,\mathrm{d}z = 2\Phi(z) - 1 = \mathrm{erf}(z/\sqrt{2})\,, \tag{B.80}$$

which can be seen to have the expansion

$$\Sigma^{(k)}(0) = \begin{cases} \frac{2}{\sqrt{\pi}} \cdot \frac{(-1)^{(k-1)/2}}{2^{k/2}\left(\frac{k-1}{2}\right)!k} & \text{if } k \text{ is odd,} \\ 0 & \text{otherwise.} \end{cases} \tag{B.81}$$

Again, the sign function is expressive since $\Sigma^{(k)} \neq 0$ for every odd k. □

B.3 Proof of Proposition 4

Using the definition of INAL and the Fourier expansion of f, we get

$$\mathrm{INAL}(f,\sigma) = \mathbb{E}_{w,b}\left[\langle f,\sigma\rangle^2\right] \tag{B.82}$$

$$= \mathbb{E}_{w,b}\left[\left(\sum_{T\in[n]} \hat{f}(T)\langle\chi_T,\sigma\rangle\right)^2\right] \tag{B.83}$$

$$= \mathbb{E}_{w,b}\left[\sum_{T} \hat{f}(T)^2\langle\chi_T,\sigma\rangle^2 + \sum_{S\neq T} \hat{f}(S)\hat{f}(T)\langle\chi_T,\sigma\rangle\langle\chi_S,\sigma\rangle\right]. \tag{B.84}$$

We show that the second term of (B.84) is zero. Let S, T be two distinct sets. Without loss of generality, assume that $|S| \geq |T|$, and let i be such that $i \in S$ but $i \notin T$ (such i must exist since $S \neq T$). Fix w and b and decompose w into $w_{\sim i}, |w_i|, \mathrm{sgn}(w_i)$ where $w_{\sim i}$ denotes the vector of weights, excluding coordinate i. By applying the change of variable $\mathrm{sgn}(w_i)x_i \mapsto y_i$ and noticing that x_i has the same distribution of y_i, we then get

$$\langle \chi_S, \sigma \rangle = \mathbb{E}_x[\chi_S(x) \cdot \sigma(x_i \,\mathrm{sgn}(w_i)|w_i| + \sum_{j \neq i} x_j w_j + b)] \tag{B.85}$$

$$= \mathrm{sgn}(w_i) \cdot \mathbb{E}_{x_{\sim i}, y_i}[\chi_{S_{\sim i}}(x) \cdot y_i \cdot \sigma(y_i|w_i| + \sum_{j \neq i} x_j w_j + b)] \tag{B.86}$$

$$:= \mathrm{sgn}(w_i) \cdot E_S, \tag{B.87}$$

where E_S does not depend on $\mathrm{sgn}(w_i)$. On the other hand,

$$\langle \chi_T, \sigma \rangle = \mathbb{E}_x[\chi_T(x) \cdot \sigma(x_i \,\mathrm{sgn}(w_i)|w_i| + \sum_{j \neq i} x_j w_j + b)] \tag{B.88}$$

$$= \mathbb{E}_{x_{\sim i}, y_i}[\chi_T(x) \cdot \sigma(y_i|w_i| + \sum_{j \neq i} x_j w_j + b)], \tag{B.89}$$

which means that $\langle \chi_T, \sigma \rangle$ does not depend on $\mathrm{sgn}(w_i)$. Thus, we get

$$\mathbb{E}_{w,b}\left[\langle \chi_T, \sigma \rangle \langle \chi_S, \sigma \rangle\right] = \mathbb{E}_{w,b}\left[\mathrm{sgn}(w_i) \cdot \langle \chi_T, \sigma \rangle \cdot E_S\right] = 0. \tag{B.90}$$

Hence,

$$\mathrm{INAL}(f, \sigma) = \sum_T \hat{f}(T)^2 \mathbb{E}_{w,b}\left[\langle \chi_T, \sigma \rangle^2\right] \tag{B.91}$$

$$= \sum_T \hat{f}(T)^2 \,\mathrm{INAL}(\chi_T, \sigma). \tag{B.92}$$

B.4 Proof of Corollary 4

Indeed, by Proposition 4 for any $f : \{\pm 1\}^d \to \mathbb{R}$ and k it holds

$$\mathrm{INAL}(f, \sigma) = \sum_T \hat{f}(T)^2 \,\mathrm{INAL}(\chi_T, \sigma) \geq W^k(f) \mathrm{INAL}(\chi_k, \sigma)\,. \tag{B.93}$$

Accordingly, if $\mathrm{INAL}(\chi_k, \sigma) = \Omega(d^{-k_0})$, we have

$$W^k(f_d) \leq \mathrm{INAL}(f_d, \sigma) \cdot O(d^{k_0})\,, \tag{B.94}$$

and then, under our assumptions, also

$$W^{\leq k}(f_d) \leq \mathrm{INAL}(f_d, \sigma) \cdot O(d^{k_0})\,. \tag{B.95}$$

For the "in particular" statement, let (f_d) be a function family with negligible $\mathrm{INAL}(f_d, \sigma)$ for a correlating σ. Let $k \in \mathbb{N}$. Since σ is correlating, the assumption $\mathrm{INAL}(\chi_{k'}, \sigma) = \Omega(d^{-k_0})$ for $k' = 0, \dots, k$ holds. Therefore, (B.95) also holds and $W^{\leq k}(f)$ is negligible. Since k was arbitrary, the function family (f_d) is high-degree.

B.5 Proof of Proposition 5

Let $f = f_d$ and let $\hat{f}$ be the Fourier coefficients of the original function f, and let $\hat{h}$ be the coefficients of the augmented function $\bar{f}$. Recall that $\bar{f} : \{\pm 1\}^D \to \{\pm 1\}$ is such that $\bar{f}(x_1, ..., x_d, x_{d+1}, ..., x_D) = f(x_1, ..., x_d)$. Thus, the Fourier coefficients of $\bar{f}$ are

$$\hat{h}(T) = \begin{cases} \hat{f}(T) & \text{if } T \subseteq [d], \\ 0 & \text{otherwise.} \end{cases} \tag{B.96}$$

Let us proceed to bounding the cross-predictability. Below we denote by π a random permutation of N elements:

$$\begin{aligned}
\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D) &= \mathbb{E}_\pi \left[\mathbb{E}_x \left[\bar{f}(x) \bar{f}(\pi(x)) \right]^2 \right] && \text{(B.97)} \\
&= \mathbb{E}_\pi \left[\left(\sum_{T \subseteq [D]} \hat{h}(T) \hat{h}(\pi(T)) \right)^2 \right] && \text{(B.98)} \\
&= \mathbb{E}_\pi \left[\left(\sum_{T \subseteq [d]} \hat{f}(T) \hat{h}(\pi(T)) \cdot \mathbb{1}(\pi(T) \subseteq [d]) \right)^2 \right] && \text{(B.99)} \\
&\overset{C.S}{\leq} \mathbb{E}_\pi \left[\left(\sum_{S \subseteq [d]} \hat{h}(\pi(S))^2 \right) \cdot \left(\sum_{T \subseteq [d]} \hat{f}(T)^2 \mathbb{1}(\pi(T) \subseteq [d]) \right) \right] && \text{(B.100)} \\
&\leq \sum_{T \subseteq [d]} \hat{f}(T)^2 \cdot \mathbb{P}_\pi(\pi(T) \subseteq [d]). && \text{(B.101)}
\end{aligned}$$

Now, for any k we have

$$\begin{aligned}
\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D) &\leq \sum_{T : |T| < k} \hat{f}(T)^2 \cdot \mathbb{P}_\pi(\pi(T) \subseteq [d]) + \sum_{T : |T| \geq k} \hat{f}(T)^2 \cdot \mathbb{P}_\pi(\pi(T) \subseteq [d]) && \text{(B.102)} \\
&\leq W^{<k}(f) + \mathbb{P}_\pi(\pi(T) \subseteq [d] \,|\, |T| = k), && \text{(B.103)}
\end{aligned}$$

where the second term in (B.103) is further bounded by (recall that $D = d^{1+\epsilon}$):

$$\mathbb{P}_\pi(\pi(T) \subseteq [d] \,|\, |T| = k) = \frac{\binom{d}{k}}{\binom{D}{k}} \leq \frac{\left(\frac{de}{k}\right)^k}{\left(\frac{D}{k}\right)^k} = e^k \frac{d^k}{D^k} = e^k d^{-\epsilon \cdot k}. \tag{B.104}$$

Accordingly, for any $k \in \mathbb{N}_{>0}$ it holds that

$$\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D) \leq W^{<k}(f) + e^k d^{-\epsilon k} . \tag{B.105}$$

Now, if (f_d) is a high degree sequence of Boolean functions, then $W^{<k}(f)$ is negligible for every k, and therefore the cross-predictability in (B.104) is $O(d^{-k})$ for every k, that is the cross-predictability is negligible as we claimed.

On the other hand, if for some c and every k it holds that $W^{\leq k}(f_d) = O(d^{k-c})$, then we can choose $k_0 := \frac{c}{1+\epsilon}$ and apply (B.105) to get $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D) = O(d^{-\frac{\epsilon}{1+\epsilon} \cdot c})$.

B.6 Proof of Theorem 4

Let σ be an expressive activation and let (f_d) be a sequence of Boolean functions with negligible $\mathrm{INAL}(f_d, \sigma)$. By Proposition 3, σ is correlating, and by Corollary 4 (f_d) is high-degree. Therefore, by Proposition 5, the cross-predictability $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D)$ is negligible.

For the more precise statement, let (f_d) be a sequence of Boolean functions with $\mathrm{INAL}(f_d, \sigma) = O(d^{-c})$. By Proposition 3, σ is 1-strongly correlating. That means that for every k we have $\mathrm{INAL}(\chi_k, \sigma) = \Omega(d^{-(k+1)})$. By Corollary 4, for every k it holds $W^{\leq k}(f_d) = O(d^{k+1-c})$. Finally, applying Proposition 5, we have that $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D) = O(d^{-\frac{\epsilon}{1+\epsilon}(c-1)})$.

B.7 Details and Proof of Corollary 3

Corollary 3 states a hardness results for learning on *fully connected neural networks with iid initialization.* This is a more specific definition than the one we gave for a neural network in Section 3.1.1. Let us state it precisely, following the treatment in Abbe and Sandon, 2020b.

Definition 30. *For the purposes of Corollary 3, a neural network on n inputs consists of a differentiable activation function $\sigma : \mathbb{R} \to \mathbb{R}$, a threshold function $f : \mathbb{R} \to \{\pm 1\}$ and a weighted, directed graph with a vertex set labeled with $\{1, x_1, \ldots, x_d, v_1, \ldots, v_m, v_{\mathrm{out}}\}$. The vertices labeled with $x_1, \ldots, x_d$ are called the input vertices, the vertex labeled with 1 is the constant vertex and v_{out} is the output vertex.*

We assume that the graph does not contain loops, the constant and input vertices do not have any incoming edges, the output vertex does not have outgoing edges and for the remaining vertices there are no edges (v_i, v_j) for $i > j$. Each vertex (a neuron) has an associated function (the output of the neuron) from $\mathbb{R}^d$ to $\mathbb{R}$ which is defined recursively as follows: The output of the constant vertex is $y_1 = 1$ and the output of the input vertex is (abusing notation) $y_{x_i} = x_i$. The output of any other vertex v_i is given by $y_{v_i} = \sigma(\sum_{v:(v,v_i)\in E(G)} w_{v,v_i} y_v)$. Finally, the output of the whole network is given by $f(y_{v_{\mathrm{out}}})$.

We say that the neural network is fully connected *if every vertex that has an incoming edge* *from an input vertex has incoming edges from all input vertices.*

Note that our definition of "fully connected network" covers any feed-forward architecture that consists of a number of fully connected hidden layers stacked on top of each other.

Let us restate Theorem 3 from Abbe and Sandon, 2020b with the bound[I] from their Corrolary 1 applied to the junk flow term JF_T:

Theorem 15 (Abbe and Sandon, 2020b)**.** *Let $P_{\mathscr{F}}$ be a distribution on Boolean functions $f : \{\pm 1\}^d \to \{\pm 1\}$. Consider any neural network as defined in Definition 30 with E edges. Assume that a function f is chosen from $P_{\mathscr{F}}$ and then T steps of noisy GD with gradient range A and noise level τ are run on the initial network using function f and uniform input distribution $\mathscr{U}^d$.*

Then, in expectation over the initial choice of f, the training noise, and a fresh sample $x \sim \mathscr{U}^d$, the trained neural network $\mathrm{NN}^{(T)}$ *satisfies*

$$\Pr\left[\mathrm{NN}(x;\theta^T) = f(x)\right] \le \frac{1}{2} + \frac{T\sqrt{P}A}{\tau} \cdot \mathrm{CP}(P_{\mathscr{F}}, \mathscr{U}^d)^{1/4} \,. \tag{B.106}$$

Finally, we need to discuss the fact that Corollary 3 applies for any fully connected neural network *with iid initialization.* What we mean by this is that the initial neural network has a fixed activation σ, threshold function f and graph (vertices and edges), but the weights on edges are not fixed. Instead, they are chosen randomly iid from any fixed probability distribution. More precisely, we can make a weaker assumption that the weights on all edges that are outgoing from the input vertices are chosen[II] iid from a fixed distribution and all the other weights have arbitrary fixed values.

We can now proceed to prove Corollary 3.

B.7.1 Proof of Corollary 3

Let a randomly initialized, fully connected neural network NN be trained in the following way. First, a function $\overline{f_d} \circ \pi$ is chosen uniformly at random from the orbit of $\overline{f_d}$. Then, a noisy GD algorithm is run with the parameters stated: T steps, learning rate γ, overflow range A and noise level τ. Finally, a fresh sample $x \sim \mathscr{U}^D$ is presented to the trained neural network. Then, Theorem 15 says that

$$\Pr\left[\mathrm{NN}(x;\theta^T) = (\overline{f_d} \circ \pi)(x)\right] \le \frac{1}{2} + \frac{T\sqrt{P}A}{\tau} \cdot \mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathscr{U}^D)^{1/4} \,. \tag{B.107}$$

[I]Since we are discussing GD, we are applying their bound with infinite sample size $m = \infty$.

[II]Even more precisely, we can assume only that the distribution of these weights is symmetric under permutations of input vertices $x_1, \ldots, x_d$.

Since we can apply Theorem 15 to the class of all orbits of $-\overline{f_d}$, which has the same cross-predictability, the same upper bound also holds for $\Pr[\mathrm{NN}(x;\theta^T) \neq (\overline{f_d} \circ \pi)(x)]$. Consequently, we have the expectation bound

$$\left|\mathbb{E}\langle \mathrm{NN}(.;\theta^T), \overline{f_d} \circ \pi\rangle\right| \leq \frac{2T\sqrt{P}A}{\tau} \cdot \mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D)^{1/4}\,. \tag{B.108}$$

Recall that the neural network is fully connected and the weights on the edges outgoing from the input vertices are iid. The expectation in (B.108) is an average of conditional expectations for different initial choices of permutation π. Consider the action induced by π on the weights outgoing from the input vertices. By properties of GD, it follows that each conditional expectation over π contributes equally to the left-hand side of (B.108). It follows that the same bound holds also for the single function $\overline{f_d}$:

$$\left|\mathbb{E}_{\theta^T}\langle \mathrm{NN}(.;\theta^T), \overline{f_d}\rangle\right| \leq \frac{2T\sqrt{P}A}{\tau} \mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D)^{1/4}\,. \tag{B.109}$$

Accordingly, if $\mathrm{INAL}(f_d, \sigma)$ is negligible, then, by Theorem 4, $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D)$ is negligible and the right-hand side of (B.109) remains negligible for any polynomial bounds on T, P, A and τ, as claimed.

For the more precise statement, if $\mathrm{INAL}(f_d, \sigma) = O(d^{-c})$, then again by Theorem 4 it holds $\mathrm{CP}(\mathrm{orb}(\overline{f_d}), \mathcal{U}^D) = O(d^{-\frac{\epsilon}{1+\epsilon}(c-1)})$ and we get the bound of $O\left(\frac{T\sqrt{P}A}{\tau} \cdot d^{-\frac{\epsilon}{4(1+\epsilon)}(c-1)}\right)$ on the right-hand side of (B.109).

C Appendix on Initial Gradient Alignment

C.1 Proofs of Theorem 6 and Theorem 7

C.1.1 Proof of Lemma 1

This proof follows a similar argument that is used in (Abbe and Boix-Adserà, 2022; Abbe and Sandon, 2020a). $\mathrm{TV}(\theta^T;\psi^T)$ can be bounded by:

$$\begin{aligned}
\mathrm{TV}(\theta^T;\psi^T) &= \mathrm{TV}\big(\theta^{T-1}-\gamma(\mathit{\Gamma}_f(\theta^{T-1})+Z^t);\psi^{T-1}-\gamma(\mathit{\Gamma}_r(\psi^{T-1})+\xi^t)\big) && \text{(C.1)}\\
&\overset{a)}{\leq} \mathrm{TV}\big(\theta^{T-1}-\gamma(\mathit{\Gamma}_f(\theta^{T-1})+Z^t);\psi^{T-1}-\gamma(\mathit{\Gamma}_f(\psi^{T-1})+Z^t)\big) && \text{(C.2)}\\
&\quad+\mathrm{TV}\big(\psi^{T-1}-\gamma(\mathit{\Gamma}_f(\psi^{T-1})+Z^t);\psi^{T-1}-\gamma(\mathit{\Gamma}_r(\psi^{T-1})+\xi^t)\big) && \text{(C.3)}\\
&\overset{b)}{\leq} \mathrm{TV}\big(\theta^{T-1};\psi^{T-1}\big)+\mathrm{TV}\big(\gamma(\mathit{\Gamma}_f(\psi^{T-1})+Z^t);\gamma(\mathit{\Gamma}_r(\psi^{T-1})+\xi^t)\big) && \text{(C.4)}\\
&\overset{c)}{\leq} \mathrm{TV}\big(\theta^{T-1};\psi^{T-1}\big)+\frac{1}{2\tau\gamma}\mathbb{E}_{\psi^{T-1}}\|\gamma\mathit{\Gamma}_f(\psi^{T-1})-\gamma\mathit{\Gamma}_r(\psi^{T-1})\|_2 && \text{(C.5)}\\
&= \mathrm{TV}\big(\theta^{T-1};\psi^{T-1}\big)+\frac{1}{2\tau}\mathbb{E}_{\psi^{T-1}}\|\mathit{\Gamma}_f(\psi^{T-1})-\mathit{\Gamma}_r(\psi^{T-1})\|_2 && \text{(C.6)}
\end{aligned}$$

where in $a)$ we used the triangular inequality, in $b)$ the data processing inequality (DPI), and in $c)$ Pinsker's inequality. Thus,

$$\begin{aligned}
\mathrm{TV}(\theta^T;\psi^T) &\leq \frac{1}{2\tau}\sum_{t=0}^{T-1}\mathbb{E}_{\psi^t}\|\mathit{\Gamma}_f(\psi^t)-\mathit{\Gamma}_r(\psi^t)\|_2 && \text{(C.7)}\\
&\overset{(a)}{\leq} \frac{1}{2\tau}\sum_{t=0}^{T-1}\sqrt{\mathbb{E}_{\psi^t}\|\mathit{\Gamma}_f(\psi^t)-\mathit{\Gamma}_r(\psi^t)\|_2^2}, && \text{(C.8)}
\end{aligned}$$

where in (a) we used the Cauchy-Schwarz inequality.

C.1.2 Proof of Proposition 6

Recall, that $\theta^0 \sim \mathcal{N}(0, V)$, where V is a $P \times P$ diagonal matrix such that $V_{pp} = v_{l_p}^2$, where l_p is the layer of parameter p, and $\psi_p^t \sim \mathcal{N}(0, U)$, where U is a $P \times P$ diagonal matrix such that $U_{pp} = v_{l_p}^2 + t\gamma^2\tau^2$. Thus, $U = \overline{C}V\overline{C}^T$, where $\overline{C}$ is a $P \times P$ diagonal matrix such that

$$\overline{C}_{pp} = \sqrt{1 + \frac{t\gamma^2\tau^2}{v_{l_p}^2}}. \tag{C.9}$$

Definition 31 ($\overline{C}$-Rescaling)**.** *Let* $\mathrm{NN}(x;\theta)$ *be an* L*-layers network, with parameters* $\theta \in \mathbb{R}^P$*. Let* $C^{(1)}, ..., C^{(L)}$ *be* L *positive constants, and let* $\overline{C}$ *be a* $P \times P$ *diagonal matrix such that* $\overline{C}_{pp} = C^{(l_p)}$ *where* l_p *is the layer of parameter* θ_p*. We say that the vector* $\overline{C} \cdot \theta$ *is a* $\overline{C}$*-rescaling of* θ*.*

Definition 32 (Strong Positive Homogeneity (SPH))**.** *We say that an architecture is* H-strongly homogeneous *(*H*-SPH) if for all* $\overline{C}$*-rescaling such that* $\min_{p\in[P]} \overline{C}_{pp} > 1$*, it holds:*

$$\mathrm{NN}(x; \overline{C}\cdot\theta) = C_{p,H} \cdot \mathrm{NN}(x;\theta), \tag{C.10}$$

$$\partial_{(\overline{C}\theta)_p} \mathrm{NN}(x; \overline{C}\cdot\theta) = D_{p,H} \cdot \partial_{\theta_p} \mathrm{NN}(x;\theta), \tag{C.11}$$

where $C_{p,H} = \prod_{l=1}^{l_p} \left(C^{(l)}\right)^H$ *and* $D_{p,H}$ *is such that* $D_{p,H} \le C_{p,H}$*.*

Lemma 9. *Let* $\mathrm{NN}(x;\theta)$ *be a fully connected network as in* (3.20)-(3.21)*. Assume that the activation* σ *is* H*-strongly homogeneous (as defined in Def. 12), with* $H \ge 1$*. Then,* $\mathrm{NN}(x;\theta)$ *is* H*-SPH.*

The proof of Lemma 9 is in Appendix C.1.3.

If we optimize over the Correlation Loss, i.e. $L(f,\theta,x) := -f(x)\cdot\mathrm{NN}(x;\theta)$, then the gradients of interest are given by:

$$\Gamma_f(\theta) = -\mathbb{E}_x\left[f(x)\cdot\nabla_\theta \mathrm{NN}(x;\theta)\right]; \tag{C.12}$$

$$\Gamma_r(\theta) = 0. \tag{C.13}$$

Thus,

$$\mathbb{E}_{\psi^t}\|\Gamma_f(\psi^t) - \Gamma_r(\psi^t)\|_2^2 = \sum_{p=1}^P \mathbb{E}_{\psi^t}\mathbb{E}_x\left[\partial_{\psi_p^t}\mathrm{NN}(x;\psi^t)\cdot f(x)\right]^2$$

Let $\overline{C}$ be a $P \times P$ matrix such that $\overline{C}_{pp} = \sqrt{1 + \frac{t\gamma^2\tau^2}{v_{l_p}^2}}$, where l_p is the layer of θ_p^0. One can verify that the $\overline{C}$-rescaling of θ^0 has the same distribution as ψ^t. We can thus rewrite each term in the sum above as:

$$\begin{aligned}\mathbb{E}_{\psi^t}\mathbb{E}_x\left[\partial_{\psi^t_p}\mathrm{NN}(x;\psi^t)\cdot f(x)\right]^2 &= \mathbb{E}_{\overline{C}\theta^0}\mathbb{E}_x\left[\partial_{(\overline{C}\theta^0)_p}\mathrm{NN}(x;\overline{C}\theta^0)\cdot f(x)\right]^2\\ &\overset{(a)}{=} D^2_{p,H}\cdot\mathbb{E}_{\theta^0}\mathbb{E}_x\left[\partial_{\theta^0_p}\mathrm{NN}(x;\theta^0)\cdot f(x)\right]^2\end{aligned}$$

where in (a) we used Lemma 9. Thus,

$$\begin{aligned}\mathbb{E}_{\psi^t}\|\Gamma_f(\psi^t)-\Gamma_r(\psi^t)\|_2^2 &= \mathbb{E}_{\theta^0}\sum_{p=1}^P D^2_{p,H}\mathbb{E}_x\left[\partial_{\theta^0_p}\mathrm{NN}(x;\theta^0)\cdot f(x)\right]^2\\ &\overset{(a)}{\le} K\cdot\mathbb{E}_{\theta^0}\|G_f(\theta^0)\|_2^2,\end{aligned}$$

where $K=\prod_{l=1}^L\left(1+\frac{t\gamma^2\tau^2}{v_l^2}\right)^H$, and where in (a) we used that $|D_{p,H}|\le C_{p,H}$.

C.1.3 Proof of Lemma 9

We proceed by induction on the network depth. As a base case, we consider a 2-layer network. Let us write explicitly the gradients of the network.

$$\nabla_{W_i^{(2)}}\mathrm{NN}(x;\theta)=\sigma(x_i^{(1)}(\theta)), \tag{C.14}$$

$$\nabla_{W_{ij}^{(1)}}\mathrm{NN}(x;\theta)=W_i^{(2)}\sigma'(x_i^{(1)}(\theta))x_j, \tag{C.15}$$

$$\nabla_{b_i^{(1)}}\mathrm{NN}(x;\theta)=W_i^{(2)}\sigma'(x_i^{(1)}(\theta)). \tag{C.16}$$

Notice that the strong homogeneity assumption on the activation σ (Def. 12), we have for $l\in\{1,2\}$:

$$x_i^{(l)}(\overline{C}\cdot\theta)=\prod_{h=1}^l(C^{(h)})^H\cdot x_i^{(l)}(\theta), \tag{C.17}$$

thus (C.10) holds. Moreover,

$$\partial_{W_i^{(2)}}\mathrm{NN}(x;\overline{C}\cdot\theta)=(C^{(1)})^H\partial_{W_i^{(2)}}\mathrm{NN}(x;\theta), \tag{C.18}$$

$$\partial_{W_{ij}^{(1)}}\mathrm{NN}(x;\overline{C}\cdot\theta)=(C^{(2)})^H\partial_{W_{ij}^{(1)}}\mathrm{NN}(x;\theta), \tag{C.19}$$

$$\partial_{b_i^{(1)}}\mathrm{NN}(x;\overline{C}\cdot\theta)=(C^{(2)})^H\partial_{b_i^{(1)}}\mathrm{NN}(x;\theta). \tag{C.20}$$

Therefore, for any parameter θ_p, $p\in[P]$,

$$\partial_{\theta_p}\mathrm{NN}(x;\overline{C}\cdot\theta)=D_{p,H}\partial_{\theta_p}\mathrm{NN}(x;\theta), \tag{C.21}$$

with $1<D_{p,H}\le\max\{(C^{(1)})^H,(C^{(2)})^H\}\le\prod_{l=1}^2(C^{(l)})^H$.

For the induction step, assume that for a network of depth $L-1$, for all parameters θ_p,

$$\partial_{\theta_p} \mathrm{NN}(x; \overline{C}(\theta)) = D_{p,H} \cdot \partial_{\theta_p} \mathrm{NN}(x; \theta), \tag{C.22}$$

with $1 < D_{p,H} \leq \prod_{l=1}^{L-1} (C^{(l)})^H$. Let us consider a neural network of depth L, and let us write the gradients,

$$\partial_{W_i^{(L)}} \mathrm{NN}(x; \theta) = \sigma(x_i^{(L-1)}(\theta)), \tag{C.23}$$

$$\partial_{W_{ij}^{(l)}} \mathrm{NN}(x; \theta) = \sum_{k=1}^{N_{L-1}} W_k^{(L)} \sigma'(x_k^{(L-1)}(\theta)) \cdot \partial_{W_{ij}^{(l)}} x_k^{(L-1)}(\theta), \qquad l = 1, \ldots, L-1, \tag{C.24}$$

$$\partial_{b_i^{(1)}} \mathrm{NN}(x; \theta) = \sum_{k=1}^{N_{L-1}} W_k^{(L)} \sigma'(x_k^{(L-1)}(\theta)) \cdot \partial_{b_i^{(1)}} x_k^{(L-1)}(\theta), \tag{C.25}$$

where N_{L-1} denotes the width of the $(L-1)$-th hidden layer. One can observe that $x_k^{(L-1)}(\theta)$ corresponds to the output of a fully connected network of depth $L-1$, and thus we can use the induction hypothesis for bounding $\partial_{\theta_p} x_k^{(L-1)}(\theta)$, for all parameters θ_p in the first $L-1$ layers. Thus,

$$\partial_{W_i^{(L)}} \mathrm{NN}(x; \overline{C}(\theta)) = (C^{(L-1)})^H \cdot D_{W_i^{(L)},H} \cdot \partial_{W_i^{(L)}} \mathrm{NN}(x; \theta), \tag{C.26}$$

$$\partial_{W_{ij}^{(l)}} \mathrm{NN}(x; \overline{C}(\theta)) = \sum_{k=1}^{N_{L-1}} C^{(L)} W_k^{(L)} \sigma'(x_k^{(L-1)}(\theta)) \cdot D_{W_{ij}^{(l)},H} \partial_{W_{ij}^{(l)}} x_k^{(L-1)}(\theta), \qquad l = 1, \ldots, L-1, \tag{C.27}$$

$$\partial_{b_i^{(1)}} \mathrm{NN}(x; \overline{C}(\theta)) = \sum_{k=1}^{N_{L-1}} C^{(L)} W_k^{(L)} \sigma'(x_k^{(L-1)}(\theta)) \cdot D_{b_i^{(1)},H} \partial_{b_i^{(l)}} x_k^{(L-1)}(\theta). \tag{C.28}$$

Thus, the result follows.

C.1.4 Proof of Proposition 7

For ease of notation, let us denote $A(\psi^t) := \|\Gamma_f(\psi^t) - \Gamma_r(\psi^t)\|_2^2$ and by $\phi_{\mathcal{N}}$ the density of a $\mathcal{N}(0, \gamma^2 t\tau^2 \mathbb{I})$ random vector. Let us write:

$$\int_{\mathbb{R}^P} A(\xi) \int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi - \psi)\phi_\mu(\psi) d\psi d\xi \tag{C.29}$$

$$= \int_{\mathbb{R}^P} A(\xi)\sqrt{\phi_\mu(\xi)} \frac{1}{\sqrt{\phi_\mu(\xi)}} \int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi - \psi)\phi_\mu(\psi) d\psi d\xi \tag{C.30}$$

$$\overset{(a)}{\le} \left(\int_{\mathbb{R}^P} A(\xi)^2 \phi_\mu(\xi) d\xi\right)^{1/2} \cdot \left(\int_{\mathbb{R}^P} \frac{1}{\phi_\mu(\xi)} \left(\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi - \psi)\phi_\mu(\psi) d\psi\right)^2 d\xi\right)^{1/2} \tag{C.31}$$

$$= \left(\int_{\mathbb{R}^P} A(\xi)^2 \phi_\mu(\xi) d\xi\right)^{1/2} \cdot K_\mu^{1/2}, \tag{C.32}$$

where in (a) we used Cauchy-Schwartz. The result follows since $A(\xi) = \|G_f(\xi)\|_2^2$.

C.1.5 Computation of K_μ for Gaussian and Uniform Distributions

Gaussian Initialization. Let us first consider the i.i.d. case. Assume that $\mu = \mathcal{N}(0; V)$, where $V \in \mathbb{R}^{P\times P}$ is a diagonal matrix such that $V_{ii} = v_i^2$.

$$K_\mu = \int_{\mathbb{R}^P} \frac{1}{\phi_\mu(\xi)} \left(\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi - \psi)\phi_\mu(\psi) d\psi d\xi\right)^2 \tag{C.33}$$

$$= \int_{\mathbb{R}^P} \frac{1}{\phi_\mu(\xi)} \phi_{\mu+\mathcal{N}}(\xi)^2 d\xi, \tag{C.34}$$

where $\mu + \mathcal{N} \sim \mathcal{N}(0, V + \gamma^2 t\tau^2 \mathbb{I})$. Thus, if for all $i \in [P]$, $v_i^2 \ge \gamma^2 t\tau^2$,

$$K_\mu = \prod_{i=1}^P \int_{\mathbb{R}} \frac{\phi_{(\mu+\mathcal{N})_i}(\xi_i)^2}{\phi_{(\mu)_i}(\xi_i)} d\xi_i \tag{C.35}$$

$$= \prod_{i=1}^P \frac{v_i}{\sqrt{2\pi(v_i^2 + \gamma^2 t\tau^2)}} \int_{\mathbb{R}} \exp\left(-\frac{\xi_i^2}{2}\left(\frac{v_i^2 - \gamma^2 t\tau^2}{v_i^2(v_i^2 + \gamma^2 t\tau^2)}\right)\right) d\xi_i \tag{C.36}$$

$$= \prod_{i=1}^P \frac{v_i}{\sqrt{2\pi(v_i^2 + \gamma^2 t\tau^2)}} \cdot \sqrt{2\pi} \sqrt{\frac{v_i^2(v_i^2 + \gamma^2 t\tau^2)}{v_i^2 - \gamma^2 t\tau^2}} \tag{C.37}$$

$$= \prod_{i=1}^P \frac{v_i^2}{\sqrt{(v_i^2 + \gamma^2 t\tau^2)(v_i^2 - \gamma^2 t\tau^2)}} \tag{C.38}$$

$$= \prod_{i=1}^P \frac{v_i^2}{\sqrt{v_i^4 - \gamma^4 t^2\tau^4}}. \tag{C.39}$$

On the other hand, in the case where V is a generic covariance matrix we have:

$$K_\mu = \int_{\mathbb{R}^P} \frac{1}{2\pi^{P/2}} \exp\left(-x^T(V+\gamma^2 t\tau^2\mathbb{I})^{-1}x + \frac{1}{2}x^T V^{-1}x\right)dx. \tag{C.40}$$

Uniform Initialization. Let us assume that $\mu = \mathrm{Unif}([-1,1])^{\otimes P}$. Let C be such that $\|\psi^t\|_\infty < C$ with probability at least $1-\exp(-d\delta)$, for some $\delta > 0$. Assume that for all $C > 0$, $A(C\cdot\xi) = C\cdot A(\xi)$ (this holds e.g. for fully connected networks with ReLU activation). Then,

$$\int_{\mathbb{R}^P} A(\xi)\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi-\psi)\phi_\mu(\psi)d\psi d\xi = \int_{[-C,C]^{\otimes P}} A(\xi)\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi-\psi)\phi_\mu(\psi)d\psi d\xi + \exp(-d\delta), \tag{C.41}$$

We apply the change of variable $\xi_i \mapsto C\cdot y_i$, for all $i\in[P]$, in the first term of the right hand side, which gives:

$$\int_{[-C,C]^{\otimes P}} A(\xi)\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(\xi-\psi)\phi_\mu(\psi)d\psi d\xi \tag{C.42}$$

$$= C^P\int_{[-1,1]^{\otimes P}} A(y)\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(y-\psi)\phi_\mu(\psi)d\psi dy \tag{C.43}$$

$$\overset{(a)}{\leq} C^P\left(\int_{[-1,1]^{\otimes P}} A(y)^2dy\right)^{1/2}\cdot\left(\int_{[-1,1]^{\otimes P}}\left(\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(y-\psi)\phi_\mu(\psi)d\psi\right)^2dy\right)^{1/2} \tag{C.44}$$

$$= \mathbb{E}_{\theta^0\sim\mu}[A(\theta^0)^2]^{1/2}\cdot K_\mu, \tag{C.45}$$

where in (a) we used Cauchy-Schwartz inequality and K_μ is given by

$$K_\mu = 2^{P/2}C^P\cdot\left(\int_{[-1,1]^{\otimes P}}\left(\int_{\mathbb{R}^P} \phi_{\mathcal{N}}(y-\psi)\phi_\mu(\psi)d\psi\right)^2dy\right)^{1/2} \tag{C.46}$$

$$= 2^{P/2}C^P\prod_{i=1}^P\frac{1}{2}\left(\int_{-1}^1\left(\int_{-1}^1 \phi_{(\mathcal{N})_i}(y_i-\psi_i)d\psi_i\right)^2dy_i\right)^{1/2} \tag{C.47}$$

$$= 2^{-P/2}C^P\prod_{i=1}^P\left(\int_{-1}^1\left(\mathbb{P}(\mathcal{N}_i<y_i+1)-\mathbb{P}(\mathcal{N}_i<y_i-1)\right)^2dy_i\right)^{1/2} \tag{C.48}$$

$$\overset{(a)}{\leq} 2^{-P/2}C^P\prod_{i=1}^P\sqrt{2}(\mathbb{P}(\mathcal{N}_i<+1)-\mathbb{P}(\mathcal{N}_i<-1)) \tag{C.49}$$

$$= C^P\prod_{i=1}^P(\mathbb{P}(\mathcal{N}_i<+1)-\mathbb{P}(\mathcal{N}_i<-1)), \tag{C.50}$$

where in (a) we used that for all $i \in [P]$, the distribution $\mathcal{N}_i$ is unimodal and symmetric, with a maximum in zero.

C.2 Proof of Proposition 8

Let us denote $\mathrm{NN}(x;\theta) = \sum_{i=1}^N a_i \sigma(w_i x + b_i)$, where $\theta = (w, b, a)$ and $\sigma := \mathrm{ReLU}$. Recalling that we denoted by $\hat{f}(S) = \mathbb{E}_{x\sim \mathrm{Unif}\{\pm 1\}^d}[f(x)\chi_S(x)]$ the Fourier coefficients of f and by $W^{\leq k}(f) = \sum_{S\subseteq[d],|S|\leq k} \hat{f}(S)^2$ the Fourier weight of f up to degree k, we can rewrite:

$$\mathbb{E}_{\theta^0} \|G_f(\theta^0)\|_2^2 = \sum_{p=1}^P \mathbb{E}_{\theta^0}\mathbb{E}_x [f(x)\partial_{\theta_p^0} \mathrm{NN}(x;\theta^0)]^2 \tag{C.51}$$

$$\overset{(a)}{=} \sum_{S\subseteq[d]} \hat{f}(S)^2 \sum_{p=1}^P \mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x)\partial_{\theta_p^0} \mathrm{NN}(x;\theta^0)]^2 \tag{C.52}$$

$$\overset{(b)}{=} W^{\leq k}(f) P C + \sum_{S:|S|>k} \hat{f}(S)^2 \sum_{p=1}^P \mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x)\partial_{\theta_p^0} \mathrm{NN}(x;\theta^0)]^2, \tag{C.53}$$

where (a) follows by a similar argument as in Proposition 4 and in (b) C is a constant such that for all p, $\mathbb{E}_{\theta^0,x}[(\partial_{\theta_p^0}\mathrm{NN}(x;\theta^0))^2] \leq C$. We use the following Lemma.

Lemma 10. *Let $S \subseteq [d]$ such that $|S| = k$. Let $g : \mathbb{R} \to \mathbb{R}$ be a function that satisfies Assumption 3. Assume that $w_1, ..., w_d, b \overset{iid}{\sim} \mathcal{N}(0, \frac{1}{d+1})$ and denote by $w = (w_1, ..., w_d)$. Then,*

$$\mathbb{E}_{w,b}\mathbb{E}_x \left[\chi_S(x)\cdot g(wx+b)\right]^2 \leq \frac{\pi^2}{6}\hat{g}(m)^2 m^2, \tag{C.54}$$

where $m := \min\{k' \geq k : \hat{g}(k') \neq 0\}$.

For simplicity we denote $w = w_1^0$, $b = b_1^0$. Thus, for all S such that $|S| > k$, we can write

$$\begin{aligned}
\sum_{p=1}^P \mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x)\partial_{\theta_p^0} \mathrm{NN}(x;\theta^0)]^2 &= \sum_{i,j=1,1}^{N,d} \mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x) x_j \sigma'(w_i x + b_i)]^2 \\
&\quad + N\mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x)\sigma(w_i x + b_i)]^2 + N\mathbb{E}_{\theta^0}\mathbb{E}_x [\chi_S(x)\sigma'(w_i x + b_i)]^2 \\
&\overset{(a)}{=} \sum_{i,j=1,1}^{N,d} \mathbb{E}_{\theta^0} V^{H-1} \mathbb{E}_x \left[\chi_S(x) x_j \sigma'\left(\frac{w_i x + b_i}{\sqrt{V}}\right)\right]^2 \\
&\quad + N V^H \mathbb{E}_{\theta^0}\mathbb{E}_x \left[\chi_S(x)\sigma\left(\frac{w_i x + b_i}{\sqrt{V}}\right)\right]^2 + N V^{H-1}\mathbb{E}_{\theta^0}\mathbb{E}_x \left[\chi_S(x)\sigma'\left(\frac{w_i x + b_i}{\sqrt{V}}\right)\right]^2 \\
&\overset{(b)}{\leq} (Nd + N) V^{H-1} \frac{\pi^2}{6} \hat{\sigma}'^2(m') m'^2 + N V^H \frac{\pi^2}{6} \hat{\sigma}^2(m) m^2
\end{aligned}$$

where in (a) we used the H-homogeneity of the activation σ and we denoted by $V = v_1\sqrt{d+1}$, where v_1^2 is the variance of the initial distribution of the first layer's weights, and in (b) we used Lemma 10 and denoted $m := \min\{k' \geq k : \hat{\sigma}(k') \neq 0\}$ and $m' := \min\{k' \geq k : \hat{\sigma}'(k') \neq 0\}$. The result follows by observing that $\sum_{S \subseteq [d]} \hat{f}(S)^2 \leq 1$.

All is left is the proof of Lemma 10.

Proof of Lemma 10. Denoting $t_x = wx + b$, one can observe that for given x, x',

$$\begin{pmatrix} t_x \\ t_{x'} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & c_{x,x'} \\ c_{x,x'} & 1 \end{pmatrix} \right). \tag{C.55}$$

where $c_{x,x'} = \frac{\langle x,x' \rangle + 1}{d+1}$. Thus, applying Lemma 14, we get

$$\mathbb{E}_{w,b}\mathbb{E}_x[\chi_S(x) g(t_x)]^2] \tag{C.56}$$
$$= \mathbb{E}_{x,x'}\left[\chi_S(x)\chi_S(x') \cdot \mathbb{E}_{w,b}[g(t_x)g(t_{x'})]\right] \tag{C.57}$$
$$= \mathbb{E}_{x,x'}\left[\chi_S(x)\chi_S(x') \cdot \sum_{h=1}^{\infty} \hat{g}(h)^2 c_{x,x'}^h\right] \tag{C.58}$$
$$= \mathbb{E}_{x,x'}\left[\chi_S(x)\chi_S(x') \cdot \sum_{h=1}^{k-1} \hat{g}(h)^2 c_{x,x'}^h\right] + \mathbb{E}_{x,x'}\left[\underbrace{\chi_S(x)\chi_S(x') \cdot \sum_{h=k}^{\infty} \hat{g}(h)^2 c_{x,x'}^h}_{:= R_{g,k}(x,x')}\right] \tag{C.59}$$
$$\overset{(a)}{=} \mathbb{E}_{x,x'}\left[\chi_S(x)\chi_S(x') \cdot R_{g,k}(x,x')\right], \tag{C.60}$$

where (a) holds by Lemma 11. One the other hand, let $m := \min\{k' \geq k : \hat{g}(k') \neq 0\}$. Then,

$$\mathbb{E}_{x,x'}\left[\chi_S(x)\chi_S(x') \cdot R_{g,k}(x,x')\right] \leq \mathbb{E}_{x,x'}\left[\left|\chi_S(x)\chi_S(x') \cdot R_{g,k}(x,x')\right|\right] \tag{C.61}$$
$$\leq \sum_{h=k}^{\infty} \hat{g}(h)^2 \tag{C.62}$$
$$\overset{(a)}{\leq} \hat{g}(m)^2 m^2 \sum_{h=k}^{\infty} \frac{1}{h^2} \tag{C.63}$$
$$\leq \frac{\pi^2}{6} \hat{g}(m)^2 m^2, \tag{C.64}$$

where (a) holds because by Assumption 3, $\frac{\hat{g}(m)^2 m^2}{\hat{g}(h)^2 h^2} \geq 1$, for all $h \geq m$ such that $\hat{g}(h) \neq 0$.

□

Lemma 11. *If $f(x) = \prod_{i=1}^{P} x_i$, for $P \in \{d-1, d\}$, then for all d*

$$\mathbb{E}_{x \sim \mathcal{U}\{\pm 1\}^d}\left[f(x) \left(\sum_{i=1}^{d} x_i + 1 \right)^k \right] = \begin{cases} 0 & \textit{if } k < P \\ P! & \textit{if } k = P. \end{cases} \tag{C.65}$$

Proof. Let us write

$$\left(\sum_{i=1}^{d} x_i + 1\right)^k = \sum_{I=(i_1,\dots,i_k)\in[d]^k} \prod_{i\in I} x_i. \tag{C.66}$$

Let us focus on a single term of the sum in (C.66) for $I=(i_1,\dots,i_k)\in[d]^k$. For $j=1,\dots,d$, let $\alpha_j=\alpha_j(I):=|\{m : i_m = j\}|$. Accordingly, we can write

$$\mathbb{E}_x\left[\prod_{i=1}^{P} x_i \left(\sum_{i=1}^{d} x_i + 1\right)^k\right] = \sum_{I=(i_1,\dots,i_k)\in[d]^k} \mathbb{E}_x\left[\prod_{i=1}^{P} x_i^{\alpha_i(I)+1} \prod_{i=P+1}^{d} x_i^{\alpha_i(I)}\right] \tag{C.67}$$

$$= \sum_{I=(i_1,\dots,i_k)\in[d]^k} \prod_{i=1}^{P} \mathbb{E}_x\left[x_i^{\alpha_i(I)+1}\right] \prod_{i=P+1}^{d} \mathbb{E}_x\left[x_i^{\alpha_i(I)}\right], \tag{C.68}$$

where we assume that $\prod_{i=P+1}^{d} \mathbb{E}_x\left[x_i^{\alpha_i(I)}\right]=1$ if $P=d$. Since $\mathbb{E}[x_i^{\alpha_i+1}]=0$ if α_i is even, for a term in (C.68) to be non-zero it is necessary that α_i is odd for every $1\le i\le P$ and α_i even for $i=P+1$. If $k<P$, then since $k=\sum_{i=1}^{d}\alpha_i$ at least one of α_i, $1\le i\le P$ must be zero, and therefore even. Consequently, each term in (C.68) is zero and it follows that $\mathbb{E}_x\left[\prod_{i=1}^{P} x_i \cdot \left(\sum_{i=1}^{d} x_i+1\right)^k\right]=0$. If $P=k=d-1$, the only non-zero terms are the ones with $\alpha_i=1$ for all $1\le i\le P$ and $\alpha_d=0$. Similarly, if $P=k=d$, the only non-zero terms have $\alpha_i=1$ for all $1\le i\le P$. In both such cases, the number of I corresponding to non-zero terms is $P!$, and for each non-zero term we have $\prod_{i=1}^{P}\mathbb{E}_x\left[x_i^{\alpha_i(I)+1}\right]\prod_{i=P+1}^{d}\mathbb{E}_x\left[x_i^{\alpha_i(I)}\right]=1$. Thus, the second line of the claim holds. □

C.3 Background on Hermite Polynomials

We present here a brief introduction to Hermite polynomials and their basic properties, and we refer e.g. to Silverman et al., 1972 for details.

C.3.1 The Hermite polynomials

The n-th (probabilist) Hermite polynomial is defined by

$$H_n(x) = \frac{(-1)^n}{\phi(x)} \frac{d^n}{dx^n}\phi(x), \tag{C.69}$$

where $\phi(x) := \frac{e^{-x^2/2}}{\sqrt{2\pi}}$. One can show (by integration by parts) that for all $n,m\in\mathbb{N}$:

$$\int_{\mathbb{R}} H_n(x)H_m(x)\phi(x)dx = n!\cdot\delta_{nm}, \tag{C.70}$$

where δ_{nm} is the Kronecker delta. Thus, the Hermite polynomials form an orthogonal basis of $L^2(\mathcal{N}(0,1))$, and every function $\sigma : \mathbb{R} \to \mathbb{R}$ in $L^2(\mathcal{N}(0,1))$, can be expanded in a series of Hermite polynomials

$$\sigma(x) = \sum_{n=0}^{\infty} \hat{\sigma}(n) H_n(x), \tag{C.71}$$

where $\hat{\sigma}(n)$ denote the *Hermite coefficients*. To compute the Hermite coefficients, one can multiply (C.71) by $\phi(x)H_m(x)$ and integrate term by term, obtaining:

$$\int_{\mathbb{R}} \sigma(x) H_m(x) \phi(x) dx = \sum_{n=0}^{\infty} \hat{\sigma}(n) \int_{\mathbb{R}} H_n(x) H_m(x) \phi(x) dx \tag{C.72}$$

$$\overset{(a)}{=} \hat{\sigma}(m) \cdot m!, \tag{C.73}$$

where in (a) we used (C.70). Thus,

$$\hat{\sigma}(n) = \frac{1}{n!} \int_{\mathbb{R}} \sigma(x) H_n(x) \phi(x) dx. \tag{C.74}$$

C.3.2 Useful lemmas

Lemma 12. *Let $H_n(x)$ be the n-th (probabilist) Hermite polynomial. Then,*

$$\int_{\mathbb{R}} H_n(x) \exp\left(-\frac{(x - cy)}{2(1-c^2)}\right) dx = \sqrt{2\pi} c^n \sqrt{1-c^2} H_n(y). \tag{C.75}$$

Note that the above corresponds to $\mathbb{E}[H_k(x)]$ for x distributed as a non-centered Gaussian distribution.

Proof. Recall, $H_n(x) = \frac{(-1)^n}{\phi(x)} \frac{d^n}{dx^n} \phi(x)$, where $\phi(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}}$. Observe,

$$\sum_{k=0}^{\infty} \frac{c^k t^k}{k!} H_k(x) = \frac{1}{\phi(x)} \sum_{k=0}^{\infty} \frac{(-ct)^k}{k!} \frac{d^k}{dx^k} \phi(x) \tag{C.76}$$

$$\overset{(a)}{=} \frac{\phi(x - ct)}{\phi(x)} = e^{ctx - \frac{(ct)^2}{2}}, \tag{C.77}$$

where in (a) we used Taylor's theorem. Thus,

$$\int_{-\infty}^{\infty} \exp\left(-\frac{(x-cy)^2}{2(1-c^2)}\right)\sum_{k=0}^{\infty}\frac{t^k}{k!}H_k(x)dx \tag{C.78}$$

$$= \int_{-\infty}^{\infty} \exp\left(-\frac{(x-cy)^2}{2(1-c^2)}\right)\exp(tx-t^2/2)dx \tag{C.79}$$

$$= \int_{-\infty}^{\infty} \exp\left(-\frac{x^2}{2(1-c^2)}+\frac{x[t(1-c^2)+cy]}{(1-c^2)}-t^2/2-\frac{c^2y^2}{2(1-c^2)}\right)dx \tag{C.80}$$

$$= \sqrt{2\pi(1-c^2)}\exp\left(cty-\frac{c^2t^2}{2}\right)dx. \tag{C.81}$$

□

Lemma 13. *Let G_1, G_2 be two jointly Gaussian variables such that* $\text{Var}(G_1) = \text{Var}(G_2) = 1$ *and* $\text{cov}(G_1, G_2) = c$. *Let $H_n(x)$ be the n-th (probabilist) Hermite polynomial. Then,*

$$\mathbb{E}_{G_1,G_2}[H_n(G_1)H_m(G_2)] = c^n\delta_{nm}, \qquad \forall n,m \in \mathbb{N}, \tag{C.82}$$

where δ_{nm} is the Kronecker delta.

Proof.

$$\mathbb{E}_{G_1,G_2}[H_n(G_1)H_m(G_2)] = \int_{\mathbb{R}^2} H_n(x)H_m(y)\frac{\exp\left(-\frac{(y-cx)}{2(1-s^2)}\right)}{\sqrt{2\pi(1-s^2)}}dxdy \tag{C.83}$$

$$\overset{(a)}{=} \int_{\mathbb{R}} H_n(x)H_m(x)c^m dx = c^n\delta_{nm}, \tag{C.84}$$

where in (a) we used Lemma 12. □

Lemma 14. *Let G_1, G_2 be two jointly Gaussian variables such that* $\text{Var}(G_1) = \text{Var}(G_2) = 1$ *and* $\text{cov}(G_1, G_2) = c$. *Let $h : \mathbb{R} \to \mathbb{R}$ be a real function with Hermite expansion given by $h(x) = \sum_{n=0}^{\infty}\hat{h}(n)H_n(x)$, where H_n is the n-th (probabilist) Hermite polynomial. Then,*

$$\mathbb{E}_{G_1,G_2}[h(G_1)h(G_2)] = \sum_{k=0}^{\infty}\hat{h}(k)^2c^k. \tag{C.85}$$

Proof. Uses Lemma 13 □

Lemma 15. *Let $h : \mathbb{R} \to \mathbb{R}$ be a real differentiable function with Hermite expansion given by $h(x) = \sum_{n=0}^{\infty}\hat{h}(n)H_n(x)$, where H_n is the n-th (probabilist) Hermite polynomial. Let $h'(x)$ be* *the first derivative of $h(x)$ and let $h'(x) = \sum_{n=0}^{\infty} \hat{h}'(n)H_n(x)$ be the Hermite expansion of h'. Then,*

$$\hat{h}'(n) = \hat{h}(n+1) \tag{C.86}$$

Proof. We use the following recurrence relation:

$$i) \qquad H_n'(x) = xH_n(x) - H_{n+1}(x), \tag{C.87}$$

Then,

$$\int_{\mathbb{R}} h'(x)H_n(x)\phi(x)dx \overset{(a)}{=} -\int_{\mathbb{R}} h(x)\big(H_n'(x) - xH_n(x)\big)\phi(x)dx \tag{C.88}$$

$$\overset{(b)}{=} \int_{\mathbb{R}} h(x)H_{n+1}(x)\phi(x)dx. \tag{C.89}$$

where in (a) we used integration by part and in (b) we used (C.87). □

C.3.3 Hermite coefficients of ReLU and threshold functions

Lemma 16 (Hermite coefficients of ReLU.)**.** *Let $\sigma(x) := \mathrm{ReLU}(x)$. Then, the coefficients of the (probabilist) Hermite expansion of σ are given by*

$$\hat{\sigma}(k) = \begin{cases} \frac{1}{2} & \text{if } k = 1\,, \\ \frac{(-1)^{k/2+1}}{\sqrt{2\pi}2^{k/2}(k-1)(k/2)!} & \text{if } k \text{ is even,} \\ 0 & \text{otherwise.} \end{cases} \tag{C.90}$$

Proof. Recall, $\hat{\sigma}(k)$ are defined by

$$\hat{\sigma}(k) = \frac{1}{k!}\int_{\mathbb{R}} \sigma(x)H_k(x)\phi(x)dx \tag{C.91}$$

$$= \frac{(-1)^k}{k!}\int_{\mathbb{R}} \sigma(x)\frac{d^k}{dx^k}\phi(x)dx \tag{C.92}$$

$$= \frac{(-1)^k}{k!}\frac{d^k}{dz^k}(\sigma * \phi)(z)\Big|_{z=0}, \tag{C.93}$$

where $*$ denotes the convolution in $\mathbb{R}$, defined as $(\sigma * \phi)(z) = \int_{\mathbb{R}} \sigma(x)\phi(z-x)dx$. Consider the Taylor expansions of $(\sigma * \phi)(z)$ around 0, i.e.

$$(\sigma * \phi)(z) = \sum_{k=0}^{\infty} \frac{1}{k!} \cdot \frac{d^k}{dz^k}(\sigma * \phi)(z)\Big|_{z=0} \cdot z^k, \tag{C.94}$$

and observe that $\hat{\sigma}(k)$ corresponds to the k-th coefficient of such expansion, up to a $(-1)^k$

term. To compute the Taylor coefficients of $(\sigma * \phi)(z)$ we write for $\sigma = \text{ReLU}$,

$$\begin{aligned}(\sigma * \phi)(z) &= \int_{\mathbb{R}} \mathbb{1}(z + y \geq 0)(z+y)\phi(y)\,\mathrm{d}y && \text{(C.95)}\\ &= \int_{-z}^{\infty} (z+y)\phi(y)\,\mathrm{d}y && \text{(C.96)}\\ &= z\Phi(z) + \phi(z) && \text{(C.97)}\\ &= \frac{z}{2} + \frac{z\,\mathrm{erf}(z/\sqrt{2})}{2} + \phi(z), && \text{(C.98)}\end{aligned}$$

where we denoted by $\Phi(z)$ the CDF of the standard gaussian distribution and by $\mathrm{erf}(z) = \frac{2}{\pi}\int_0^z e^{-t^2}\,dt$ the Gaussian error function. Using well known Taylor expansions of erf and ϕ we get the result. □

Lemma 17 (Hermite coefficients of threshold function.)**.** *Let $\sigma(x) := \mathbb{1}(x > 0)$. Then, the coefficients of the (probabilist) Hermite expansion of σ are given by*

$$\hat{\sigma}(k) = \begin{cases} \frac{1}{\sqrt{2\pi}} \cdot \frac{(-1)^{(k+1)/2}}{2^{\frac{k-1}{2}} k \left(\frac{k-1}{2}\right)!} & \textit{if } k \textit{ is odd,} \\ 0 & \textit{otherwise.} \end{cases} \tag{C.99}$$

Proof. Using a similar strategy as in the Proof of Lemma 16, we write for $\sigma = \mathbb{1}(x > 0)$

$$\begin{aligned}(\sigma * \phi)(z) &= \int_{\mathbb{R}} \mathbb{1}(z + y \geq 0)\phi(y)\,\mathrm{d}y && \text{(C.100)}\\ &= \int_{-z}^{\infty} \phi(y)dy && \text{(C.101)}\\ &= \Phi(z) && \text{(C.102)}\\ &= \frac{1}{2} + \mathrm{erf}(z/\sqrt{2}). && \text{(C.103)}\end{aligned}$$

Using the well known Taylor expansion of erf we get the result. □

D Appendix on Noise Stability

D.1 Proof of Theorem 8

The proof of Theorem 8 goes by two steps. Firstly, we connect the noise stability to another measure of complexity for classes of functions called cross predictability (CP), defined in Definition 8. Then, similarly to the proof of Theorem 4, we use the negative results from (Abbe and Sandon, 2020a), that lower bound the generalization error of learning a class of functions in terms of its cross predictability.

Recall that for a function $f : \{\pm 1\}^d \to \mathbb{R}$ we defined its $2d$-extension as $\overline{f} : \{\pm 1\}^{2d} \to \mathbb{R}$ as $\overline{f}(x_1, ..., x_d, x_{d+1}, ..., x_{2d}) = f(x_1, ..., x_d)$ (see Def. 7), and its orbit as the class containing all functions formed by composing f with any permutation of the input coordinates (Def. 9). Recall the Cross-Predictability (CP) defined in Def. 8. For brevity we make use of the following notation:

$$\mathrm{CP}(\mathrm{orb}(\overline{f})) := \mathrm{CP}(\mathcal{U}_{\mathrm{orb}(\overline{f})}, \mathcal{U}^{2d}), \tag{D.1}$$

where $\mathcal{U}_{\mathrm{orb}(\overline{f})}, \mathcal{U}^{2d}$ denote the uniform distribution over $\mathrm{orb}(\overline{f})$ and over $\{\pm 1\}^{2d}$, respectively.

Furthermore, recall that every Boolean function f can be written in terms of its Fourier-Walsh expansion $f(x) = \sum_S \hat{f}(S)\chi_S(x)$, where $\chi_S(x) = \prod_{i \in S} x_i$ are the standard Fourier basis elements and $\hat{f}(S)$ are the Fourier coefficients of f. We further denote by

$$W^k(f) = \sum_{S:|S|=k} \hat{f}(S)^2 \quad \text{and} \quad W^{\le k}(f) = \sum_{S:|S|\le k} \hat{f}(S)^2, \tag{D.2}$$

the total weight of the Fourier coefficients of f at degree k and up to degree k, respectively. We denote by $\hat{f}$ the Fourier coefficients of the original function f, and by $\hat{h}$ the coefficients of the augmented function $\overline{f}$, that are:

$$\hat{h}(T) = \hat{f}(T) \qquad \text{if } T \subseteq [d] \tag{D.3}$$

$$\hat{h}(T) = 0 \qquad \text{otherwise.} \tag{D.4}$$

Remark 17 (Noise Stability)**.** *We remark the following two properties of* $\mathrm{Stab}_\delta[f]$*:*

1. *One can show (see e.g., Theorem 2.49 in O'Donnell, 2014) that*

$$\mathrm{Stab}_\delta(f) = \sum_{k=1}^{d} (1-2\delta)^k W^k[f], \tag{D.5}$$

   where $W^k[f]$ *is the Boolean weight at degree* k *of* f*;*

2. *For all* $\delta \in [0, 1/2]$, $\mathrm{Stab}_\delta(f) = \mathrm{Stab}_\delta(\overline{f})$*. This follows directly from the previous point and* (D.3)-(D.4)*.*

We make use of the following Lemma, that relates the cross-predictability of $\mathrm{orb}(\overline{f})$ to the Stability of f.

Lemma 18. *There exists* δ *such that for any* $\delta' < \delta$

$$\mathrm{CP}(\mathrm{orb}(\overline{f})) \le \mathrm{Stab}_{\delta'}(f). \tag{D.6}$$

Proof. We denote by π a random permutation of $2d$ elements. Following a similar technique as in the proof of Proposition 5, we can bound the $\mathrm{CP}(\mathrm{orb}(\overline{f}))$ by the following:

$$\mathrm{CP}(\mathrm{orb}(\overline{f})) = \mathbb{E}_\pi \left[\mathbb{E}_x \left[\overline{f}(x) \overline{f}(\pi(x)) \right]^2 \right] \tag{D.7}$$

$$= \mathbb{E}_\pi \left(\sum_{T \subseteq [2d]} \hat{h}(T) \hat{h}(\pi(T)) \right)^2 \tag{D.8}$$

$$= \mathbb{E}_\pi \left(\sum_{T \subseteq [d]} \hat{f}(T) \hat{f}(\pi(T)) \cdot \mathbb{1}(\pi(T) \subseteq [d]) \right)^2 \tag{D.9}$$

$$\overset{C.S}{\le} \mathbb{E}_\pi \left(\sum_{S \subseteq [d]} \hat{f}(\pi(S))^2 \right) \cdot \left(\sum_{T \subseteq [d]} \hat{f}(T)^2 \mathbb{1}(\pi(T) \subseteq [d]) \right) \tag{D.10}$$

$$= \sum_{T \subseteq [d]} \hat{f}(T)^2 \cdot \mathbb{P}_\pi(\pi(T) \subseteq [d]) \tag{D.11}$$

$$= \sum_{k=1}^{d} W^k[f] \cdot \mathbb{P}_\pi(\pi(T) \subseteq [d] \,||T| = k), \tag{D.12}$$

where (D.8) is the scalar product in the Fourier basis, (D.9) follows by applying the formulas of the $\hat{h}$ given in (D.3)-(D.4), (D.10) holds by Cauchy-Schwarz inequality, (D.11) holds since

f is Boolean-valued and for each π by Parseval identity, $\sum_{S\subseteq[d]}\hat{f}(\pi(S))^2 = \mathbb{E}_x[f(x)^2] = 1$, and (D.12) holds since the second term is invariant for all sets of a given cardinality. Recalling π is a random permutation over the augmented input space of dimension $2d$, for each $k \in [d]$ we can further bound the second term by

$$\mathbb{P}_\pi(\pi(T)\subseteq[d]\,|\,|T|=k)=\frac{\binom{d}{k}}{\binom{2d}{k}}\sim\frac{1}{2^k}\leq(1-2\delta')^k,\quad \text{for all } \delta'\leq 1/4. \tag{D.13}$$

Thus, for all $\delta' \leq 1/4$,

$$\mathrm{CP}(\mathrm{orb}(\overline{f}))\leq\sum_{k=1}^{d}(1-2\delta')^k W^k[f]=\mathrm{Stab}_{\delta'}[f]. \tag{D.14}$$

□

Remark 18. *Note that the value of δ in Lemma 18 depends on the size of the input extension that we use. In this paper, we defined an input extension of size $2d$ (input doubling), which gives $\delta = 1/4$, however we could have chosen e.g. a $3d$-extension and obtain $\delta = 1/3$, and so on.*

For the second step, we make use of Theorem 3 and Corollary 1 in (Abbe and Sandon, 2020a) (see Thm. 5), that prove a lower bound of learning a class of function in terms of its cross predictability. The lower bound holds for the noisy GD algorithm, defined in (1.9). Theorem 3 and Corollary 1 in (Abbe and Sandon, 2020a) imply that for any distribution over the Boolean hypercube $P_{\mathscr{X}}$ and Boolean functions $P_{\mathscr{F}}$, it holds that

$$\mathbb{P}_{x\sim P_{\mathscr{X}},F\sim P_{\mathscr{F}},\theta^T}(F(x)\neq\mathrm{NN}(x;\theta^T))\geq 1/2-\frac{T\sqrt{P}A}{\tau}(1/B+\mathrm{CP}(P_{\mathscr{F}},P_{\mathscr{X}}))^{1/4}, \tag{D.15}$$

where γ, P, A, τ, B have the same meaning as in Definition (1.9). In our case since the initialization is invariant under permutations of the input, then learning the orbit of $\overline{f}$ under uniform distribution is equivalent to learning $\overline{f}$, thus the following bound holds:

$$\mathbb{P}_{x,\theta^T}(\overline{f}(x)\neq\mathrm{NN}(x;\theta^T))\geq 1/2-\frac{T\sqrt{P}A}{\tau}\left(1/B+\mathrm{CP}(\mathrm{orb}(\overline{f}))\right)^{1/4}. \tag{D.16}$$

D.2 Removing the input doubling

One can prove a similar result to the one of Theorem 8, without using the input extension technique. However, we need some additional assumptions on f. To introduce them, let us first fix some notation. In the following, we say that a sequence a_n is *noticeable* if there exists $c \in \mathbb{N}$ such that $a_n = \Omega(n^{-c})$. On the other hand, we say that f is *negligible* if $\lim_{n\to\infty} n^c a_n = 0$ for every $c \in \mathbb{N}$ (which we also write $a_n = n^{-\omega(1)}$).

Assumption 4 (Non-dense and non-extremal function). *a) We say that f is "non-dense" if there exists c such that $W\{T : \hat{f}(T)^2 \leq d^{-c}\} = d^{-\omega(1)}$, i.e. the negligible Fourier coefficients do not bring a noticeable contribution if taken all together;*

b) We say f is "non-extremal" if for any positive constant D, $W^{\geq d-D}[f] = d^{-\omega(1)}$, i.e. f does not have noticeable Fourier weight on terms of degree $d - O(1)$.

With such additional assumptions, we can conclude the following.

Proposition 15. *Let $f : \{\pm 1\}^d \to \{\pm 1\}$ be a balanced target function, let $\mathrm{Stab}_\delta(f)$ be its noise stability and let $\mathrm{NN}(x;\theta^t)$ be the output of noisy-GD with gradient range A and noise level τ after t time steps, trained on a neural network of size P with initialization that is invariant to permutation of the input neurons. Assume f satisfies Assumption 4. Then, there exist $c, C > 0$ and $D > 0$ such that if $\delta < D/d$*

$$\mathbb{P}_{x,\theta^t}(f(x) \neq \mathrm{NN}(x;\theta^t)) \geq 1/2 - \frac{Ct\sqrt{P}}{\tau} \cdot \left(d^c \cdot \mathrm{Stab}_\delta(f) + d^{-\omega(1)}\right)^{1/4}. \tag{D.17}$$

The proof of Proposition 15 resembles the proof of Theorem 8. The only modification required is in Lemma 18, which is replaced by the following Lemma.

Lemma 19. *Let f be a Boolean function that satisfies Assumption 4. There exists $c, D > 0$ such that for $\delta < D/n$,*

$$\mathrm{CP}(\mathrm{orb}(f)) \leq 2 \cdot d^c \cdot \mathrm{Stab}_\delta(f) + d^{-\omega(1)}. \tag{D.18}$$

Proof. Let $c > 0$ be such that $W\{T : \hat{f}(T)^2 \leq d^{-c}\} = d^{-\omega(1)}$. This c exists because of Assumption 1a.

$$\mathrm{CP}(\mathrm{orb}(f)) = \tag{D.19}$$
$$= \mathbb{E}_\pi\left[\mathbb{E}_x\left[f(x)f(\pi(x))\right]^2\right] \tag{D.20}$$
$$= \mathbb{E}_\pi\left(\sum_{T\subseteq[d]} \hat{f}(T)\hat{f}(\pi(T))\right)^2 \tag{D.21}$$
$$= \mathbb{E}_\pi\left(\sum_{T\subseteq[d]} \hat{f}(T)\hat{f}(\pi(T)) \cdot \left(\mathbb{1}\left(\hat{f}(\pi(T))^2 \leq d^{-c}\right) + \mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right)\right)\right)^2 \tag{D.22}$$
$$\leq 2\mathbb{E}_\pi\left(\sum_{T\subseteq[d]} \hat{f}(T)\hat{f}(\pi(T))\mathbb{1}\left(\hat{f}(\pi(T))^2 \leq d^{-c}\right)\right)^2 + \tag{D.23}$$
$$+ 2\mathbb{E}_\pi\left(\sum_{T\subseteq[d]} \hat{f}(T)\hat{f}(\pi(T))\mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right)\right)^2, \tag{D.24}$$

where in the last inequality we used $(a+b)^2 \leq 2(a^2+b^2)$. Let us first focus on the second term on the right.

$$\mathbb{E}_\pi \left(\sum_{T \subseteq [d]} \hat{f}(T)\hat{f}(\pi(T))\mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right)\right)^2 \tag{D.25}$$

$$\overset{C.S}{\leq} \mathbb{E}_\pi \left(\sum_{S \subseteq [d]} \hat{f}(\pi(S))^2 \right) \cdot \left(\sum_{T \subseteq [d]} \hat{f}(T)^2 \mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right)\right) \tag{D.26}$$

$$\leq \sum_{T \subseteq [d]} \hat{f}(T)^2 \cdot \mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c}\right) \tag{D.27}$$

$$= \sum_{k=1}^{n} W^k[f] \cdot \mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c} \,||T| = k\right) \tag{D.28}$$

$$= \sum_{k=1}^{d-D} W^k[f] \cdot \mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c} \,||T| = k\right) + \tag{D.29}$$

$$+ \sum_{k=d-D+1}^{d} W^k[f] \cdot \mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c} \,||T| = k\right)$$

$$\leq \sum_{k=1}^{d-D} W^k[f] \cdot \mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c} \,||T| = k\right) + W^{\geq d-D+1}[f]. \tag{D.30}$$

where D is an arbitrary positive constant. Because of Assumption 1b, $W^{\geq d-D+1}[f] = d^{-\omega(1)}$. On the other hand, since f is a Boolean valued function,

$$\sum_T \hat{f}(T)^2 = \mathbb{E}_x[f(x)^2] = 1, \tag{D.31}$$

which implies that there are at most d^c sets T such that $\hat{f}(T)^2 > d^{-c}$. Thus, recalling π is a random permutation over the input space of dimension d, we get

$$\mathbb{P}_\pi \left(\hat{f}(\pi(T))^2 > d^{-c} \,||T| = k\right) \leq \frac{d^c}{\binom{d}{k}} \tag{D.32}$$

$$\leq d^c \left(\frac{k}{d}\right)^k \tag{D.33}$$

$$\leq d^c \left(\frac{d-D}{d}\right)^k \tag{D.34}$$

$$\leq d^c (1-2\delta)^k \qquad \text{if } \delta \leq \frac{D}{2d}, \tag{D.35}$$

where in (D.33) we used that $\binom{d}{k} \geq (\frac{d}{k})^k$ for all $k \geq 1$. Going back to the first term in (D.24) we get

$$\mathbb{E}_\pi \left(\sum_{T\subseteq[d]} \hat{f}(T)\hat{f}(\pi(T))\mathbb{1}\left(\hat{f}(\pi(T))^2 \le d^{-c}\right)\right)^2 \tag{D.36}$$

$$\overset{C.S}{\le} \mathbb{E}_\pi \left(\sum_{S\subseteq[d]} \hat{f}(S)^2 \right) \cdot \left(\sum_{T\subseteq[d]} \hat{f}(\pi(T))^2 \mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right)\right) \tag{D.37}$$

$$\le \sum_{T\subseteq[d]} \hat{f}(\pi(T))^2 \mathbb{1}\left(\hat{f}(\pi(T))^2 > d^{-c}\right) \tag{D.38}$$

$$= d^{-\omega(1)}, \tag{D.39}$$

by Assumption 1a. Hence overall,

$$\mathrm{CP}(\mathrm{orb}(f)) \le 2d^c \sum_{k=1}^{d-D} W^k[f](1-2\delta)^k + d^{-\omega(1)} \tag{D.40}$$

$$\le 2d^c\, \mathrm{Stab}_\delta(f) + d^{-\omega(1)}. \tag{D.41}$$

□

D.3 Noise Stability of PVR functions

As mentioned in Section 1.2.1, a Pointer-Value-Retrieval (PVR) task consists of a pointer (the first bits of the input) and an aggregation function that acts on a specific window indicated by the pointer. We denote by p the number of bits that define the pointer, and by w the size of each window. For simplicity, we consider a slight variation of Boolean PVR task with non-overlapping windows, defined as follows:

- PVR with *non-overlapping* windows: the 2^p windows pointed by the pointer bits are non-overlapping, i.e., the first window is formed by bits $x_{p+1},...,x_{p+w}$, the second window is formed by bits $x_{p+w+1},...,x_{p+2w}$, and so forth.

The input size is thus given by $d := p+2^p w$ and $p = O(\log(d))$. We denote by $g : \{\pm 1\}^w \to \{\pm 1\}$ the aggregation function, which we assume to be balanced (i.e. $\mathbb{E}_x[g(x)] = 0$). One can verify (see details below) that the noise stability of the PVR function f is given by

$$\mathrm{Stab}_\delta[f] = (1-\delta)^{p+w} + (1-\delta)^p(1-(1-\delta)^w)\cdot \mathrm{Stab}_\delta[g]. \tag{D.42}$$

We notice that the $\mathrm{Stab}_\delta[f]$ is given by two terms: the first one depends on the window size and the second one on the stability of the aggregation function. For large enough window size, the second term in (D.42) is the dominant one, and $\mathrm{Stab}_\delta[f]$ depends on the stability of g. Thus from Theorem 8, f is not learned by GD (in the extended input space) in poly(n) time if the stability of the aggregation function is $d^{-\omega(1)}$. On the other hand, for small window size (specifically for $w = O(\log(d))$), the $\mathrm{Stab}_\delta(f)$ is 'noticeable' (as defined in Appendix D.2) for every aggregation function, since the function value itself depends on a limited number of input bits. Thus, noise unstable aggregation functions (e.g. parities) can form a PVR function with 'noticeable' stability, if the window size is $O(\log(d))$.

Computation of (D.42)**.** We compute the expression in (D.42) with the following:

$$\mathrm{Stab}_\delta[f] = 1 - 2\,\mathrm{NS}_\delta[f], \tag{D.43}$$

where $\mathrm{NS}_\delta[f] := \mathbb{P}(f(x) \neq f(y))$ is the Noise sensitivity of f, defined as the probability that perturbing each input bit independently with probability δ changes the output of f and where we denoted by y the vector obtained from x by flipping each component with prob. δ independently. To compute $\mathrm{NS}_\delta[f]$, we can first distinguish depending on whether the perturbation affects the pointer bit:

$$\begin{aligned}
\mathrm{NS}_\delta[f] &:= \mathbb{P}(f(x) \neq f(y)) \\
&= (1-\delta)^p \cdot \mathbb{P}(f(x) \neq f(y) \mid x^p = y^p) + (1-(1-\delta)^p)\mathbb{P}(f(x) \neq f(y) \mid x^p \neq y^p) \\
&= (1-\delta)^p \cdot \mathbb{P}(f(x) \neq f(y) \mid x^p = y^p) + (1-(1-\delta)^p)\frac{1}{2},
\end{aligned}$$

where the last inequality holds since we are using non-overlapping windows and we assumed g to be balanced. To compute the first term, we can condition on whether any bit in the window pointed by x and y is changed:

$$\begin{aligned}
\mathbb{P}(f(x) &\neq f(y) \mid x^p = y^p) \\
&= (1-\delta)^w \cdot \mathbb{P}(f(x) \neq f(y) \mid x^p = y^p, X_{P(x^p)} = Y_{P(Y^p)}) + \\
&\qquad + (1-(1-\delta)^w) \cdot \mathbb{P}(f(x) \neq f(y) \mid x^p = y^p, x_{P(x^p)} \neq y_{P(y^p)}) \\
&= (1-(1-\delta)^w) \cdot \mathbb{P}(f(x) \neq f(y) \mid x^p = y^p, x_{P(x^p)} \neq y_{P(y^p)}) \\
&= (1-(1-\delta)^w) \cdot \mathrm{NS}_\delta[g],
\end{aligned}$$

where the last inequality holds because g is unbalanced. By replacing $\mathrm{NS}_\delta[g] = \frac{1}{2} - \frac{1}{2}\,\mathrm{Stab}_\delta[g]$ and rearranging terms one can obtain (D.42).

E Appendix on Boolean Influence

In this section, we present proofs for results mentioned in Section **??**, namely, Lemma 2 and Theorem 9.

E.1 Proof of Lemma 2

Proof of Lemma 2. Let $f(x) = \sum_{T\subseteq[n]} \hat{f}(T)\chi_T(x)$ be the Fourier expansion of the function where $\chi_T(x) = \prod_{i\in T} x_i$. Therefore, the Fourier expansion of the frozen function will become

$$f_{-k}(x) = \sum_{T\subseteq[n]\setminus k} (\hat{f}(T) + \hat{f}(T\cup k))\chi_T(x). \tag{E.1}$$

Thus, the difference between functions is equal to

$$(f - f_{-k})(x) = \sum_{T\subseteq[n]:k\in T} \hat{f}(T)\chi_T(x) - \sum_{T\subseteq[n]\setminus k} \hat{f}(T\cup k)\chi_T(x). \tag{E.2}$$

Hence, using Parseval's Theorem we have the following:

$$\mathbb{E}_{U^n}(f - f_{-k})_2^2 = \sum_{T\subseteq[n]:k\in T} \hat{f}(T)^2 + \sum_{T\subseteq[n]\setminus k} \hat{f}(T\cup k)^2 = 2\sum_{T\subseteq[n]:k\in T} \hat{f}(T)^2. \tag{E.3}$$

Therefore,

$$\mathbb{E}_{U^n}\frac{1}{2}(f - f_{-k})_2^2 = \sum_{T\subseteq[n]:k\in T} \hat{f}(T)^2 = \mathrm{Inf}_k(f), \tag{E.4}$$

and the lemma is proved. □

E.2 Proof of Theorem 9

Proof of Theorem 9. Assume $\tilde{f}_{-k}^{(t)}(x,\Theta^{(t)}) := x^T W^{(t)} + b^{(t)}$ to be our linear model where $\Theta^{(t)} = (W^{(t)}, b^{(t)})$ are the model parameters at time t. In the following, the super-script t and T

denote the time-step and transpose respectively. Also, we use $\mathbb{E}_{x_{-k}}$ to denote the expectation of x taken uniformly on the Boolean hypercube while $x_k = 1$. Using the square loss, we have

$$L(\Theta^{(t)}, x, f) = (x^T W^{(t)} + b^{(t)} - f(x))^2, \tag{E.5}$$

and the gradients will be

$$\nabla_W L(\Theta^{(t)}, x, f) = 2x\left(x^T W^{(t)} + b^{(t)} - f(x)\right), \tag{E.6}$$
$$\partial_b L(\Theta^{(t)}, x, f) = 2\left(x^T W^{(t)} + b^{(t)} - f(x)\right). \tag{E.7}$$

The GD update rule will then become

$$W^{(t+1)} = W^{(t)} - 2\gamma\left(\mathbb{E}_{x_{-k}}\left[xx^T\right]W^{(t)} + \mathbb{E}_{x_{-k}}[x]b^{(t)} - \mathbb{E}_{x_{-k}}[xf(x)]\right), \tag{E.8}$$
$$b^{(t+1)} = b^{(t)} - 2\gamma\left(\mathbb{E}_{x_{-k}}[x^T]W^{(t)} + b^{(t)} - \mathbb{E}_{x_{-k}}[f(x)]\right). \tag{E.9}$$

Note that $\mathbb{E}_{x_{-k}}\left[xx^T\right] = \mathbb{I}_n$, $\mathbb{E}_{x_{-k}}[x] = \vec{e}_k$. So we have

$$\forall j \neq k: \; W_j^{(t+1)} = W_j^{(t)}(1-2\gamma) + 2\gamma\mathbb{E}_{x_{-k}}[x_j \cdot f(x)], \tag{E.10}$$
$$W_k^{(t+1)} = W_k^{(t)} - 2\gamma(W_k^{(t)} + b^{(t)}) + 2\gamma\mathbb{E}_{x_{-k}}[f(x)], \tag{E.11}$$
$$b^{(t+1)} = b^{(t)} - 2\gamma(W_k^{(t)} + b^{(t)}) + 2\gamma\mathbb{E}_{x_{-k}}[f(x)]. \tag{E.12}$$

Using above equations, we have

$$W_k^{(t+1)} - b^{(t+1)} = W_k^{(t)} - b^{(t)} = W_k^{(0)} - b^{(0)}, \tag{E.13}$$
$$W_k^{(t+1)} + b^{(t+1)} = (1-4\gamma)(W_k^{(t)} + b^{(t)}) + 4\gamma\mathbb{E}_{x_{-k}}[f(x)]. \tag{E.14}$$

Assume $\gamma < \frac{1}{4}$ and define $0 < c = -\log(1-2\gamma) < -\log(1-4\gamma)$, then we have

$$\begin{aligned} W_k^{(t)} + b^{(t)} &= (1-4\gamma)^t(W_k^{(0)} + b^{(0)} - \mathbb{E}_{x_{-k}}[f(x)]) + \mathbb{E}_{x_{-k}}[f(x)] \\ &= O((1-4\gamma)^t) + \mathbb{E}_{x_{-k}}[f(x)] = O(e^{-ct}) + \mathbb{E}_{x_{-k}}[f(x)] \\ &= O(e^{-ct}) + \hat{f}(\emptyset) + \hat{f}(\{k\}), \end{aligned} \tag{E.15}$$

$$\begin{aligned} \forall j \neq k: \; W_j^{(t)} &= (1-2\gamma)^t(W_j^{(0)} - \mathbb{E}_{x_{-k}}[x_j \cdot f(x)]) + \mathbb{E}_{x_{-k}}[x_j \cdot f(x)] \\ &= O((1-2\gamma)^t) + \mathbb{E}_{x_{-k}}[x_j \cdot f(x)] = O(e^{-ct}) + \mathbb{E}_{x_{-k}}[x_j \cdot f(x)] \\ &= O(e^{-ct}) + \hat{f}(\{j\}). \end{aligned} \tag{E.16}$$

So the learned function is

$$\begin{aligned} \tilde{f}_{-k}(x; \Theta^{(t)}) = \frac{b^{(0)} - W_k^{(0)} + \hat{f}(\emptyset) + \hat{f}(\{k\})}{2} &+ \frac{W_k^{(0)} - b^{(0)} + \hat{f}(\emptyset) + \hat{f}(\{k\})}{2} x_k \\ &+ \sum_{j \neq k} \hat{f}(\{j\}) \cdot x_j + O(e^{-ct}) \end{aligned} \tag{E.17}$$

and the generalization error can be computed using Parseval Theorem:

$$\mathrm{gen}(f,\tilde{f}_{-k}^{(t)}) = \frac{1}{2}\mathbb{E}_{x\sim U^n}\left[\left(f(x)-\tilde{f}_{-k}^{(t)}(x;\Theta^{\infty})\right)^2\right] \tag{E.18}$$

$$= \frac{1}{2}\left(\frac{(b^{(0)}-W_k^{(0)}-\hat{f}(\emptyset)+\hat{f}(\{k\}))^2+(W_k^{(0)}-b^{(0)}+\hat{f}(\emptyset)-\hat{f}(\{k\}))^2}{4}\right)+O(e^{-ct}) \tag{E.19}$$

$$= \frac{(b^{(0)}-W_k^{(0)}-\hat{f}(\emptyset)+\hat{f}(\{k\}))^2}{4}+O(e^{-ct}) \tag{E.20}$$

$$= \frac{(b^{(0)}-W_k^{(0)})^2}{4}+\frac{(\hat{f}(\emptyset)-\hat{f}(\{k\}))^2}{4}-2\frac{(b^{(0)}-W_k^{(0)})(\hat{f}(\emptyset)-\hat{f}(\{k\})}{4}+O(e^{-ct}). \tag{E.21}$$

Therefore, the expected generalization loss over different initializations is given by

$$\mathbb{E}_{\Theta^0}[\mathrm{gen}(f,\tilde{f}_{-k}^{(t)})] = \mathbb{E}_{\Theta^0}\left[\frac{(b^{(0)}-W_k^{(0)})^2+(\hat{f}(\emptyset)-\hat{f}(\{k\}))^2}{4}\right]+O(e^{-ct}) \tag{E.22}$$

$$= \frac{(\hat{f}(\emptyset)-\hat{f}(\{k\}))^2}{4}+\frac{\sigma^2}{2}+O(e^{-ct}). \tag{E.23}$$

Particularly, if the frozen function is unbiased, i.e., $\hat{f}(\emptyset)+\hat{f}(\{k\})=0$, we have

$$\begin{aligned}\mathbb{E}_{\Theta^0}[\mathrm{gen}(f,\tilde{f}_{-k}^{(t)})] &= \frac{(2\hat{f}(\{k\}))^2}{4}+\frac{\sigma^2}{2}+O(e^{-ct})\\ &= \hat{f}(\{k\})^2+\frac{\sigma^2}{2}+O(e^{-ct}) = \mathrm{Inf}_k(f)+\frac{\sigma^2}{2}+O(e^{-ct}).\end{aligned} \tag{E.24}$$

□

F Appendix on Curriculum Learning

F.1 Proof of Theorem 12

Theorem 16 (Theorem 12, restatement)**.** *Let k, d be both even integers, such that $k \leq d/2$. Let* $\mathrm{NN}(x;\theta) = \sum_{i=1}^{N} a_i \sigma(w_i x + b_i)$ *be a 2-layers fully connected network with activation* $\sigma(y) := \mathrm{Ramp}(y)$ *(as defined in* (F.2)*) and* $N \geq (d+1)(d-k+1)\log((d+1)(d-k+1)/\delta)$*. Consider training* $\mathrm{NN}(x;\theta)$ *with SGD on the hinge loss with batch size* $B \geq (8\zeta^2 N^2)^{-1}\log(\frac{Nd+N}{\delta})$*, with* $\zeta \leq \frac{\epsilon\mu^k}{24(d+1)^2(d-k+1)^2 N}$ *and* $\mu = \sqrt{1-\frac{1}{2(d-k)}}$*. Then, there exists an initialization, a learning rate schedule, and a 2-CL strategy such that after* $T \geq \frac{64}{\epsilon^2}(d-k+1)^3(d+1)N$ *iterations, with probability* $1-3\delta$ *SGD outputs a network with generalization error at most* ϵ*.*

F.1.1 Proof setup

We consider a 2-layers neural network, defined as:

$$\mathrm{NN}(x;\theta) = \sum_{i=1}^{N} a_i \sigma(w_i x + b_i), \tag{F.1}$$

where N is the number of hidden units, $\theta = (a, b, w)$ and $\sigma := \mathrm{Ramp}$ denotes the activation defined as:

$$\mathrm{Ramp}(x) = \begin{cases} 0 & x \leq 0, \\ x & 0 < x \leq 1, \\ 1 & x > 1 \end{cases}. \tag{F.2}$$

Without loss of generality, we assume that the labels are generated by $\chi_{[k]}(x) := \prod_{i=1}^{k} x_i$. Indeed, SGD on fully connected networks with permutation-invariant initialization is invariant to permutation of the input neurons, thus our result will hold for all $\chi_S(x)$ such that $|S| = k$. Our proof scheme is the following:

1. We train only the first layer of the network for one step on data $(x_i, \chi_{[k]}(x_i))_{i\in[B]}$ with $x_i \sim \mathrm{Rad}(p)^{\otimes d}$ for $i \in [B]$, with $p = \frac{1}{2}\sqrt{1-\frac{1}{2(d-k)}} + \frac{1}{2}$;

2. We show that after one step of training on such biased distribution, the target parity belongs to the linear span of the hidden units of the network;

3. We subsequently train only the second layer of the network on $(x_i, \chi_{[k]}(x_i))_{i\in[B]}$ with $x_i \sim \mathrm{Rad}(1/2)^{\otimes d}$ for $i \in [B]$, until convergence;

4. We use established results on convergence of SGD on convex losses to conclude.

We train our network with SGD on the hinge loss. Specifically, we apply the following updates, for all $t \in \{0,1,\ldots,T-1\}$:

$$\begin{aligned}
w_{i,j}^{t+1} &= w_{i,j}^t - \gamma_t \frac{1}{B}\sum_{s=1}^{B} \nabla_{w_{i,j}^t} L(\theta^t, \chi_{[k]}, x_s^t),\\
a_i^{t+1} &= a_i^t - \xi_t \frac{1}{B}\sum_{s=1}^{B} \nabla_{a_i^t} L(\theta^t, \chi_{[k]}, x_s^t) + c_t, \\
b_i^{t+1} &= \lambda_t \left(b_i^t + \psi_t \frac{1}{B}\sum_{s=1}^{B} \nabla_{b_i^t} L(\theta^t, \chi_{[k]}, x_s^t)\right) + d_t,
\end{aligned} \tag{F.3}$$

where $L(\theta^t, \chi_{[k]}, x) = \max\{0, 1-\chi_{[k]}(x)\mathrm{NN}(x;\theta^t)\}$. Following the 2-steps curriculum strategy introduced above, we set

$$x_s^0 \overset{iid}{\sim} \mathrm{Rad}(p)^{\otimes d} \qquad \forall s \in [B], \tag{F.4}$$

$$x_s^t \overset{iid}{\sim} \mathrm{Rad}(1/2)^{\otimes d} \qquad \forall t \geq 1, s \in [B], \tag{F.5}$$

where $p = \frac{1}{2}\sqrt{1-\frac{1}{2(d-k)}} + \frac{1}{2}$. For brevity, we denote $\mu := 2p-1 = \sqrt{1-\frac{1}{2(d-k)}}$. We set the parameters of SGD to:

$$\gamma_0 = \mu^{-(k-1)}2N, \qquad \gamma_t = 0 \qquad \forall t \geq 1, \tag{F.6}$$

$$\xi_0 = 0, \qquad \xi_t = \frac{\epsilon}{2N} \qquad \forall t \geq 1, \tag{F.7}$$

$$\psi_0 = \frac{N}{\mu^k}, \qquad \psi_t = 0 \qquad \forall t \geq 1, \tag{F.8}$$

$$c_0 = -\frac{1}{2N}, \qquad c_t = 0 \qquad \forall t \geq 1, \tag{F.9}$$

$$\lambda_0 = (d+1), \qquad \lambda_t = 1 \qquad \forall t \geq 1, \tag{F.10}$$

$$d_0 = 0, \qquad d_t = 0 \qquad \forall t \geq 1, \tag{F.11}$$

and we consider the following initialization scheme:

$$\begin{aligned}
&w_{i,j}^0 = 0 \qquad \forall i \in [N], j \in [d];\\
&a_i^0 = \frac{1}{2N} \qquad \forall i \in [N];\\
&b_i^0 \sim \mathrm{Unif}\left\{\frac{b_{lm}}{d+1} + \frac{1}{2} : l \in \{0, ..., d\}, m \in \{-1, ..., d-k\}\right\},
\end{aligned} \tag{F.12}$$

where we define

$$b_{lm} := -d + 2l - \frac{1}{2} + \frac{m+1}{d-k}. \tag{F.13}$$

Note that such initialization is invariant to permutations of the input neurons. We choose such initialization because it is convenient for our proof technique. We believe that the argument may generalize to more standard initialization (e.g. uniform, Gaussian), however this would require more work and it may not be a trivial extension.

F.1.2 First step: Recovering the support

As mentioned above, we train our network for one step on $(x_i, \chi_{[k]}(x_i))_{i\in[B]}$ with $x_i \sim \mathrm{Rad}(p)^{\otimes d}$.

Population gradient at initialization. Let us compute the population gradient at initialization. Since we set $\xi_0 = 0$, we do not need to compute the initial gradient for a. Note that at initialization $|\mathrm{NN}(x;\theta^0)| < 1$. Thus, the initial population gradients are given by

$$\forall j \in [k], i \in [N] \qquad G_{w_{i,j}} = -a_i \mathbb{E}_{x\sim\mathrm{Rad}(p)^{\otimes d}}\left[\prod_{l\in[k]\setminus j} x_l \cdot \mathbb{1}(\langle w_i, x\rangle + b_i \in [0,1])\right] \tag{F.14}$$

$$\forall j \notin [k], i \in [N] \qquad G_{w_{i,j}} = -a_i \mathbb{E}_{x\sim\mathrm{Rad}(p)^{\otimes d}}\left[\prod_{l\in[k]\cup j} x_l \cdot \mathbb{1}(\langle w_i, x\rangle + b_i \in [0,1])\right] \tag{F.15}$$

$$\forall i \in [N] \qquad G_{b_i} = -a_i \mathbb{E}_{x\sim\mathrm{Rad}(p)^{\otimes d}}\left[\prod_{l\in[k]} x_l \cdot \mathbb{1}(\langle w_i, x\rangle + b_i \in [0,1])\right] \tag{F.16}$$

Lemma 20. *Initialize a, b, w according to* (F.12). *Then,*

$$\forall j \in [k], \qquad G_{w_{i,j}} = -\frac{\mu^{k-1}}{2N}; \tag{F.17}$$

$$\forall j \notin [k], \qquad G_{w_{i,j}} = -\frac{\mu^{k+1}}{2N}; \tag{F.18}$$

$$G_{b_i} = -\frac{\mu^k}{2N}. \tag{F.19}$$

Proof. If we initialize according to (F.12), we have $\langle w_i, x\rangle + b_i \in [0,1]$ for all i. The results holds since $\mathbb{E}_{x\sim \mathrm{Rad}(p)^{\otimes d}}[\chi_S(x)] = \mu^{|S|}$. □

Effective gradient at initialization.

Lemma 21. *Let*

$$\hat{G}_{w_{i,j}} := \frac{1}{B}\sum_{s=1}^{B} \nabla_{w_{i,j}^0} L(\theta^0, \chi_{[k]}, x_s^t) \tag{F.20}$$

$$\hat{G}_{b_i} := \frac{1}{B}\sum_{s=1}^{B} \nabla_{b_i^0} L(\theta^0, \chi_{[k]}, x_s^t) \tag{F.21}$$

be the effective gradients at initialization. If $B \geq (8\zeta^2 N^2)^{-1}\log(\frac{Nd+N}{\delta})$*, then with probability* $1-2\delta$*,*

$$\|\hat{G}_{w_{i,j}} - G_{w_{i,j}}\|_\infty \leq \zeta, \tag{F.22}$$

$$\|\hat{G}_{b_i} - G_{b_i}\|_\infty \leq \zeta, \tag{F.23}$$

where $G_{w_{i,j}}, G_{b_i}$ *are the population gradients.*

Proof. We note that $\mathbb{E}[\hat{G}_{w_{i,j}}] = G_{w_{i,j}}$, $\mathbb{E}[\hat{G}_{b_i}] = G_{b_i}$, and $|\hat{G}_{w_{i,j}}|, |\hat{G}_{b_i}| \leq \frac{1}{2N}$

$$\mathbb{P}_{x\sim \mathrm{Rad}(p)^{\otimes d}}\left(|\,\hat{G}_{w_{i,j}} - G_{w_{i,j}}\,| \geq \zeta\right) \leq 2\exp\left(-8\zeta^2 N^2 B\right) \leq \frac{2\delta}{Nd+N}, \tag{F.24}$$

$$\mathbb{P}_{x\sim \mathrm{Rad}(p)^{\otimes d}}\left(|\,\hat{G}_{b_i} - G_{b_i}\,| \geq \zeta\right) \leq 2\exp\left(-8\zeta^2 N^2 B\right) \leq \frac{2\delta}{Nd+N}. \tag{F.25}$$

The result follows by union bound. □

Lemma 22. *Let*

$$w_{i,j}^1 = w_{i,j}^0 - \gamma_0 \hat{G}_{w_{i,j}} \tag{F.26}$$

$$b_i^1 = \lambda_0\left(b_i^0 - \psi_0 \hat{G}_{b_i}\right) \tag{F.27}$$

$$\tag{F.28}$$

If $B \geq (8\zeta^2 N^2)^{-1}\log(\frac{Nd+N}{\delta})$*, with probability* $1-2\delta$

i) For all $j \in [k]$*,* $i \in [N]$*,* $|w_{i,j}^1 - 1| \leq \frac{2N\zeta}{\mu^{k-1}}$*;*

ii) For all $j \notin [k]$*,* $|w_{i,j}^1 - (1 - \frac{1}{2(d-k)})| \leq \frac{2N\zeta}{\mu^{k-1}}$ *;*

iii) For all $i \in [N]$*,* $|b_i^1 - (d+1)(b_i^0 - \frac{1}{2})| \leq \frac{N(d+1)\zeta}{\mu^k}$*.*

Proof. We apply Lemma 22:

i) For all $j \in [k]$, $i \in [N]$, $|\hat{w}^1_{i,j} - 1| = \gamma_0 |\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \le \frac{2N\zeta}{\mu^{k-1}}$;

ii) For all $j \notin [k]$, $i \in [N]$, $|\hat{w}^1_{i,j} - (1 - \frac{1}{2(d-k)})| = \gamma_0 |\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \le \frac{2N\zeta}{\mu^{k-1}}$;

iii) For all $i \in [N]$,

$$|\hat{b}^1_i - (d+1)(b^0_i - \frac{1}{2})| = |\lambda_0(b^0_i + \psi_0 \hat{G}_{b_i}) - \lambda_0(b^0_i + \psi_0 G_{b_i})| \tag{F.29}$$
$$\le |\lambda_0| \cdot |\psi_0| \cdot |\hat{G}_{b_i} - G_{b_i}| \tag{F.30}$$
$$\le \frac{N(d+1)\zeta}{\mu^k}. \tag{F.31}$$

□

Lemma 23. *If $N \ge (d+1)(d-k+1)\log((d+1)(d-k+1)/\delta)$, then with probability $1-\delta$, for all $l \in \{0,...,d\}$, and for all $m \in \{-1,...,d-k\}$ there exists i such that $b^0_i = \frac{b_{lm}}{d+1} + \frac{1}{2}$.*

Proof. The probability that there exist l, m such that the above does not hold is

$$\left(1 - \frac{1}{(d+1)(d-k+1)}\right)^N \le \exp\left(-\frac{N}{(d+1)(d-k+1)}\right) \le \frac{\delta}{(d+1)(d-k+1)}. \tag{F.32}$$

The result follows by union bound. □

Lemma 24. *Let $\sigma_{lm}(x) = \mathrm{Ramp}\left(\sum_{j=1}^d x_j - \frac{1}{2(d-k)} \sum_{j>k} x_j + b_{lm}\right)$, with b_{lm} given in* (F.13). *If $B \ge (8\zeta^2 N^2)^{-1} \log(\frac{Nd+N}{\delta})$ and $N \ge (d+1)(d-k+1)\log((d+1)(d-k+1)/\delta)$, with probability $1-3\delta$, for all l, m there exists i such that*

$$\left|\sigma_{lm}(x) - \mathrm{Ramp}\left(\sum_{j=1}^d \hat{w}^1_{i,j} x_j + \hat{b}^1_i\right)\right| \le 3N(d+1)\zeta\mu^{-k}. \tag{F.33}$$

Proof. By Lemma 23, with probability $1-\delta$, for all l, m there exists i such that $b^0_i = \frac{b_{lm}}{d+1} + \frac{1}{2}$. For ease of notation, we replace indices $i \mapsto (l\,m)$, and denote $\hat{\sigma}_{lm}(x) = \mathrm{Ramp}\left(\sum_{j=1}^d w^1_{lm,j} x_j + b^1_{lm}\right)$. Then, by Lemma 22 with probability $1-2\delta$,

$$|\sigma_{lm}(x) - \hat{\sigma}_{lm}(x)| \le \left|\sum_{j=1}^k (w^1_{lm,j} - 1)x_j + \sum_{j=k+1}^d \left(w^1_{lm,j} - \left(1 - \frac{1}{2(d-k)}\right)\right) x_j + b^1_{lm} - b_{lm}\right| \tag{F.34}$$
$$\le k2N\zeta\mu^{-(k-1)} + (d-k)2N\zeta\mu^{-(k-1)} + N(d+1)\zeta\mu^k \tag{F.35}$$
$$\le 3N(d+1)\zeta\mu^{-k}. \tag{F.36}$$

□

Lemma 25. *There exists a^* with $\|a^*\|_\infty \le 4(d-k)$ such that*

$$\sum_{l=0}^{d}\sum_{m=-1}^{d-k} a^*_{lm}\sigma_{lm}(x) = \chi_{[k]}(x). \tag{F.37}$$

Proof. Recall, that we assumed d, k even and recall that

$$\sigma_{lm}(x) = \mathrm{Ramp}\left(\sum_{j=1}^{d} x_j - \frac{1}{2(d-k)}\sum_{j>k} x_j + b_{lm}\right), \tag{F.38}$$

where $b_{lm} = -d + 2l - \frac{1}{2} + \frac{m+1}{d-k}$ for $l \in [d], m \in \{-1, ..., d-k+1\}$ and $\mathrm{Ramp}(x) = \begin{cases} 0 & x \le 0, \\ x & 0 < x \le 1, \\ 1 & x > 1 \end{cases}$.

Let $\sum_{j=1}^{d} x_j = d - 2t$, where t is the total number of -1, and similarly let $-\sum_{j=k+1}^{d} x_j = (d-k) - 2s$, where s is the number of $+1$ outside the support of the parity $\chi_{[k]}(x)$.We have,

$$\sigma_{lm}(x) = \mathrm{Ramp}\left(2(l-t) + \frac{m+1-s}{d-k}\right). \tag{F.39}$$

We take

$$a^*_{lm} = (-1)^l(-1)^m 2(d-k) \qquad \forall l \in [d], m = -1, \tag{F.40}$$
$$a^*_{lm} = (-1)^l(-1)^m 4(d-k) \qquad \forall l \in [d], m \in \{0, 1, ..., d-k-2\}, \tag{F.41}$$
$$a^*_{lm} = (-1)^l(-1)^m 3(d-k) \qquad \forall l \in [d], m = d-k-1, \tag{F.42}$$
$$a^*_{lm} = (-1)^l(-1)^m (d-k) \qquad \forall l \in [d], m = d-k, \tag{F.43}$$

Note that for all $l < t$,

$$2(l-t) + \frac{m+1-s}{d-k} \le -2 + \frac{d-k+1}{d-k} \le 0, \tag{F.44}$$

thus, $\sigma_{lm}(x) = 0$ for all m. Moreover, for all $l > t$,

$$2(l-t) + \frac{m-s+1}{d-k} \ge 2 - \frac{d-k}{d-k} = 1. \tag{F.45}$$

Thus, $\sigma_{lm}(x) = 1$ for all m and

$$\sum_{m=-1}^{d-k} a^*_{lm}\sigma_{lm}(x) = \sum_{m=-1}^{d-k} a^*_{lm} = 0. \tag{F.46}$$

If $l = t$,

$$\begin{aligned}
&\sum_{m=-1}^{d-k} a^*_{tm}\sigma_{tm}(x)\\
&\quad = (-1)^t(d-k)\left[\sum_{m=0}^{d-k-2} 4(-1)^m \operatorname{Ramp}\left(\frac{m+1-s}{d-k}\right) - 3\operatorname{Ramp}\left(\frac{d-k-s}{d-k}\right) + \operatorname{Ramp}\left(\frac{d-k+1-s}{d-k}\right)\right]\\
&\quad = (-1)^t\left[\sum_{m=s}^{d-k-2} 4(-1)^m (m+1-s)_+ - 3(d-k-s)_+ + (d-k+1-s)_+\right]\\
&\quad = (-1)^t(-1)^s.
\end{aligned}$$

Since we assumed d, k even, $(-1)^s = \prod_{i=k+1}^d x_i$. Moreover, observe that $\chi_{[k]}(x) = \prod_{i=k+1}^d x_i \cdot \prod_{i=1}^d x_i$. Thus,

$$\sum_{lm} a^*_{lm}\sigma_{lm}(x) = (-1)^t(-1)^s = \chi_{[k]}(x). \tag{F.47}$$

□

Lemma 26. *Let $f^*(x) = \sum_{l,m} a^*_{lm}\sigma_{lm}(x)$ and let $\hat{f}(x) = \sum_{l,m} a^*_{lm}\hat{\sigma}_{lm}(x)$, with $\sigma_{lm}, \hat{\sigma}_{lm}$ defined in Lemma 30 and a^* defined in Lemma 25. If $B \geq (8\zeta^2 N^2)^{-1}\log(\frac{Nd+N}{\delta})$ and $N \geq (d+1)(d-k+1)\log((d+1)(d-k+1)/\delta)$, with probability $1-3\delta$ for all x,*

$$L(\hat{f}, f^*, x) \leq (d+1)^2(d-k+1)^2 12N\zeta\mu^{-k}. \tag{F.48}$$

Proof.

$$\begin{aligned}
|f^*(x) - \hat{f}(x)| &= \left|\sum_{l,m} a^*_{l,m}(\sigma_{lm}(x) - \hat{\sigma}_{lm}(x)\right| && \text{(F.49)}\\
&\leq d(d-k+1)\|a^*\|_\infty \sup_{lm}|\sigma_{lm}(x) - \hat{\sigma}_{lm}(x)| && \text{(F.50)}\\
&\leq (d+1)^2(d-k+1)^2 12N\zeta\mu^{-k}. && \text{(F.51)}
\end{aligned}$$

Thus,

$$\begin{aligned}
(1 - f(x)f^*(x))_+ &\leq |1 - f(x)f^*(x)| && \text{(F.52)}\\
&= |f^{*^2}(x) - f(x)f^*(x)| && \text{(F.53)}\\
&= |f^*(x)| \cdot |f^*(x) - f(x)| \leq (d+1)^2(d-k+1)^2 12N\zeta\mu^{-k}, && \text{(F.54)}
\end{aligned}$$

which implies the result. □

F.1.3 Second step: Convergence

To conclude, we use an established result on convergence of SGD on convex losses (see e.g. (Barak et al., 2022; Daniely and Malach, 2020; Malach and Shalev-Shwartz, 2020; Shalev-Shwartz et al., 2012; Shalev-Shwartz and Ben-David, 2014)).

Theorem 17. *Let $\mathscr{L}$ be a convex function and let $a^* \in \operatorname{argmin}_{\|a\|_2 \le \mathscr{B}} \mathscr{L}(a)$, for some $\mathscr{B} > 0$. For all t, let α^t be such that $\mathbb{E}\left[\alpha^t \mid a^t\right] = -\nabla_{a^t} \mathscr{L}(a^t)$ and assume $\|\alpha^t\|_2 \le \rho$ for some $\rho > 0$. If $a^0 = 0$ and for all $t \in [T]$ $a^{t+1} = a^t + \gamma \alpha^t$, with $\gamma = \frac{\mathscr{B}}{\rho\sqrt{T}}$, then :*

$$\frac{1}{T}\sum_{t=1}^{T} \mathscr{L}(a^t) \le \mathscr{L}(a^*) + \frac{\mathscr{B}\rho}{\sqrt{T}}. \tag{F.55}$$

Let $\mathscr{L}(a) := \mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[L((a, b^1, w^1), \chi_{[k]}, x)\right]$. Then, $\mathscr{L}$ is convex in a and for all $t \in [T]$,

$$\alpha^t = -\frac{1}{B}\sum_{s=1}^{B} \nabla_{a^t} L((a^t, b^1, w^1), \chi_{[k]}, x) \tag{F.56}$$

$$= -\frac{1}{B}\sum_{s=1}^{B} \sigma(w^1 x + b^1). \tag{F.57}$$

Thus, recalling $\sigma = \mathrm{Ramp}$, we have $\|\alpha^{(t)}\|_2 \le \sqrt{N}$. Let a^* be as in Lemma 25. Clearly, $\|a^*\|_2 \le 4(d-k+1)^{3/2}(d+1)^{1/2}$. Moreover, $a^1 = 0$. Thus, we can apply Theorem 17 with $\mathscr{B} = 4(d-k+1)^{3/2}(d+1)^{1/2}$, $\rho = \sqrt{N}$ and obtain that if

1. $N \ge (d+1)(d-k+1)\log((d+1)(d-k+1)/\delta)$;
2. $\zeta \le \frac{\epsilon\mu^k}{24(d+1)^2(d-k+1)^2 N}$;
3. $B \ge (8\zeta^2 N^2)^{-1}\log(\frac{Nd+N}{\delta})$;
4. $T \ge \frac{64}{\epsilon^2}(d-k+1)^3(d+1)N$.

then, with probability $1-3\delta$ over the initialization

$$\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[\min_{t\in[T]} L\left(\theta^t, \chi_{[k]}, x\right)\right] \le \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon. \tag{F.58}$$

Remark 19. *We assume $k \le d/2$ to avoid exponential dependence of ζ (and consequently of the batch size and of the computational complexity) in d. Indeed, if $k \le d/2$, then,*

$$\mu^k = \left(1 - \frac{1}{2(d-k)}\right)^{k/2} \ge \left(1 - \frac{1}{d}\right)^{d/4} \sim e^{-1/4}. \tag{F.59}$$

Remark 20 (Noisy-GD)**.** *We remark that the proof extends to the noisy-GD algorithm with* *noise level* $\tau \leq \frac{\zeta}{\sqrt{2}} \cdot \log\left(\frac{Nd+n}{\delta}\right)^{-1/2}$ *(see Def. 26). Indeed, one can replace Lemma 21 with the following Lemma, and keep the rest of the proof unchanged.*

Lemma 27 (Noisy-GD)**.** *Let*

$$\hat{G}_{w_{i,j}} := G_{w_{i,j}} + Z_{w_{i,j}}, \tag{F.60}$$

$$\hat{G}_{b_i} := G_{b_i} + Z_{b_i}, \tag{F.61}$$

be the effective gradients at initialization, where $G_{w_{i,j}}, G_{b_i}$ *are the population gradients at initialization and* $Z_{w_{i,j}}, Z_{b_i}$ *are iid* $\mathcal{N}(0,\tau^2)$*. If* $\tau \leq \frac{\zeta}{\sqrt{2}} \cdot \log\left(\frac{Nd+n}{\delta}\right)^{-1/2}$*, with probability* $1-2\delta$*:*

$$\|\hat{G}_{w_{i,j}} - G_{w_{i,j}}\|_\infty \leq \zeta, \tag{F.62}$$

$$\|\hat{G}_{b_i} - G_{b_i}\|_\infty \leq \zeta. \tag{F.63}$$

Proof. By concentration of Gaussian random variables:

$$\mathbb{P}\left(|\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \geq \zeta\right) \leq 2\exp\left(-\frac{\zeta^2}{2\tau^2}\right) \leq \frac{2\delta}{Nd+N}, \tag{F.64}$$

$$\mathbb{P}\left(|\hat{G}_{b_i} - G_{b_i}| \geq \zeta\right) \leq 2\exp\left(-\frac{\zeta^2}{2\tau^2}\right) \leq \frac{2\delta}{Nd+N}. \tag{F.65}$$

The result follows by union bound. □

F.2 Proof of Theorem 13

Theorem 18 (Theorem 13, restatement)**.** *Let* k, d *be integers, such that* $d \geq k$ *and* k *even. Let* $\mathrm{NN}(x;\theta) = \sum_{i=1}^N a_i \sigma(w_i x + b_i)$ *be a 2-layers fully connected network with activation* $\sigma(y) := \mathrm{ReLU}(y)$ *and* $N \geq (k+1)\log(\frac{k+1}{\delta})$*. Consider training* $\mathrm{NN}(x;\theta)$ *with SGD on the covariance loss with batch size* $B \geq (2\zeta^2)^{-1}\log(\frac{dN}{\delta})$*, with* $\zeta \leq \frac{\epsilon(\mu^{k-1}-\mu^{k+1})}{64k^2N} \cdot \left(1+\frac{d-k}{k}\right)^{-1}$*, for some* $\mu \in (0,1)$*. Then, there exists an initialization, a learning rate schedule, and a 2-CL strategy such that after* $T \geq \frac{64k^3N}{\epsilon^2}$ *iterations, with probability* $1-3\delta$ *SGD outputs a network with generalization error at most* ϵ*.*

F.2.1 Proof setup

Similarly as before, we consider a 2-layers neural network, defined as $\mathrm{NN}(x;\theta) = \sum_{i=1}^N a_i \sigma(w_i x + b_i)$, where N is the number of hidden units, $\theta = (a,b,w)$ and $\sigma := \mathrm{ReLU}$. Our proof scheme is similar to the previous Section. Again, we assume without loss of generality that the labels are generated by $\chi_{[k]}(x) := \prod_{i=1}^k x_i$. We assume k to be even. We train our network with SGD on the covariance loss, defined in Def. 25. We use the same updates as in (F.3) with:

$$x_s^0 \overset{iid}{\sim} \mathrm{Rad}(p)^{\otimes d} \qquad \forall s \in [B], \tag{F.66}$$

$$x_s^t \overset{iid}{\sim} \mathrm{Rad}(p)^{\otimes d} \qquad \forall t \geq 1, s \in [B], \tag{F.67}$$

for some $p \in (1/2, 1)$. We denote $\mu := 2p - 1$. We set the parameters to:

$$\gamma_0 = 16N(\mu^{k-1} - \mu^{k+1})^{-1} k^{-1}, \qquad \gamma_t = 0 \quad \forall t \geq 1, \tag{F.68}$$

$$\xi_0 = 0, \qquad \xi_t = \frac{\epsilon}{8N} \quad \forall t \geq 1, \tag{F.69}$$

$$\psi_0 = 0, \qquad \psi_t = 0 \quad \forall t \geq 1, \tag{F.70}$$

$$c_0 = -1, \qquad c_t = 0 \quad \forall t \geq 1, \tag{F.71}$$

$$\lambda_0 = 1, \qquad \lambda_t = 1 \quad \forall t \geq 1, \tag{F.72}$$

$$d_0 = -1, \qquad d_t = 0 \quad \forall t \geq 1, \tag{F.73}$$

and we consider the following initialization scheme:

$$\begin{aligned} w_{i,j}^0 &= 0 \qquad \forall i \in [N], j \in [d]; \\ a_i^0 &= \frac{1}{16N} \qquad \forall i \in [N]; \\ b_i^0 &\sim \mathrm{Unif}\left\{ \frac{2(i+1)}{k} : i \in \{0, 1, ..., k\} \right\}. \end{aligned} \tag{F.74}$$

F.2.2 First step: Recovering the support

Population gradient at initialization. At initialization, we have $|\mathrm{NN}(x; \theta^0)| < \frac{1}{4}$, thus

$$\left| \mathrm{cov}(\chi_{[k]}, \theta^0, x, \mathrm{Rad}(p)^{\otimes d}) \right| < 1. \tag{F.75}$$

The initial gradients are therefore given by:

$$\forall i, j \qquad G_{w_{i,j}} = -a_i \mathbb{E}_{x \sim \mathrm{Rad}(p)^{\otimes d}} \left[\left(\prod_{l \in [k]} x_l - \mu^k \right) \cdot \left(x_j \mathbb{1}(\langle w_i, x \rangle + b_i > 0) - \mathbb{E} x_j \mathbb{1}(\langle w_i, x \rangle + b_i > 0) \right) \right] \tag{F.76}$$

$$\forall i \in [N] \qquad G_{b_i} = -a_i \mathbb{E}_{x \sim \mathrm{Rad}(p)^{\otimes d}} \left[\left(\prod_{l \in [k]} x_l - \mu^k \right) \cdot \left(\mathbb{1}(\langle w_i, x \rangle + b_i > 0) - \mathbb{E} \mathbb{1}(\langle w_i, x \rangle + b_i > 0) \right) \right] \tag{F.77}$$

If we initialize a, b, w according to (F.74). Then,

$$\forall j \in [k], \qquad G_{w_{i,j}} = -\frac{\mu^{k-1} - \mu^{k+1}}{16N}; \tag{F.78}$$

$$\forall j \notin [k], \qquad G_{w_{i,j}} = 0; \tag{F.79}$$

$$G_{b_i} = 0. \tag{F.80}$$

Effective gradient at initialization.

Lemma 28. *Let*

$$w_{i,j}^1 = w_{i,j}^0 - \gamma_0 \hat{G}_{w_{i,j}} \tag{F.81}$$

$$b_i^1 = \lambda_0 \big(b_i^0 + \psi_0 \hat{G}_{b_i}\big) + d_0, \tag{F.82}$$

$$\tag{F.83}$$

where $\hat{G}_{w_{i,j}} := \frac{1}{B}\sum_{s=1}^B \nabla_{w_{i,j}^0} L(\theta^0, \chi_{[k]}, x_s^t)$ *and* $\hat{G}_{b_i} := \frac{1}{B}\sum_{s=1}^B \nabla_{b_i^0} L(\theta^0, \chi_{[k]}, x_s^t)$ *are the gradients estimated from the initial batch. Then, with probability* $1-2\delta$*, if* $B \ge (2\zeta^2)^{-1}\log\big(\frac{dN}{\delta}\big)$*,*

i) For all $j \in [k]$, $i \in [N]$, $|w_{i,j}^{(1)} - \frac{1}{k}| \le \frac{\zeta 16N}{k(\mu^{k-1}-\mu^{k+1})}$;

ii) For all $j \notin [k]$, $|w_{i,j}^{(1)}| \le \frac{\zeta 16N}{k(\mu^{k-1}-\mu^{k+1})}$;

iii) For all $i \in [N]$, $b_i^{(1)} = b^{(0)} - 1$

Proof. By Lemma 22 ,if $B \ge (2\zeta^2)^{-1}\log\big(\frac{dk}{\delta}\big)$, then for all $j \in [k], i \in [N]$, $|\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \le \zeta$. Thus,

i) For all $j \in [k]$, $i \in [N]$, $|w_{i,j}^{(1)} - \frac{1}{k}| = \gamma_0 |\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \le \frac{\zeta 16N}{k(\mu^{k-1}-\mu^{k+1})}$;

ii) For all $j \in [k]$, $i \in [N]$, $|w_{i,j}^{(1)}| = \gamma_0 |\hat{G}_{w_{i,j}} - G_{w_{i,j}}| \le \frac{\zeta 16N}{k(\mu^{k-1}-\mu^{k+1})}$;

iii) follows trivially. □

Lemma 29. *If* $N \ge (k+1)\log\big(\frac{k+1}{\delta}\big)$*, with probability* $1-\delta$ *for all* $i \in \{0, ..., k\}$ *there exists* $l \in [N]$ *such that* $b_l^0 = \frac{2(i+1)}{k}$*.*

Proof. The probability that there exists i such that the above does not hold is

$$\left(1 - \frac{1}{k+1}\right)^N \le \exp\left(-\frac{N}{k+1}\right) \le \frac{\delta}{k+1}. \tag{F.84}$$

The result follows by union bound. □

Lemma 30. *Let* $\sigma_i(x) := \mathrm{ReLU}\left(\frac{1}{k}\sum_{j=1}^{k} x_j + b_i\right)$*, with* $b_i = -1 + \frac{2(i+1)}{k}$*. Then, with probability* $1-3\delta$*, for all* $i \in \{0,...,k\}$ *there exists* $l \in [N]$ *such that*

$$\left|\sigma_i(x) - \mathrm{ReLU}\left(\sum_{j=1}^{d} w_{l,j}^1 x_j + b_l^1\right)\right| \le \frac{\zeta 16N}{\mu^{k-1}-\mu^{k+1}} \cdot \left(1 + \frac{d-k}{k}\right). \tag{F.85}$$

Proof. By Lemma 29 and Lemma 28, with probability $1-3\delta$, for all i there exists l such that $b_l^1 = -1 + \frac{2(i+1)}{k}$, and

$$\begin{aligned}\left|\sigma_i(x) - \mathrm{ReLU}\left(\sum_{j=1}^{d} w_{l,j}^1 x_j + b_l^1\right)\right| &\le \left|\sum_{j=1}^{k}\left(\frac{1}{k} - w_{l,j}^1 x_j\right)\right| + \left|\sum_{j=k+1}^{d} w_{l,j}^1 x_j\right| &(F.86)\\ &\le \frac{\zeta 16N}{\mu^{k-1}-\mu^{k+1}} + \frac{\zeta 16N(d-k)}{(\mu^{k-1}-\mu^{k+1})k} &(F.87)\\ &= \frac{\zeta 16N}{\mu^{k-1}-\mu^{k+1}} \cdot \left(1+\frac{d-k}{k}\right). &(F.88)\end{aligned}$$

□

Lemma 31. *There exist* a^* *with* $\|a^*\|_\infty \le 2k$ *such that*

$$\sum_{i=0}^{k} a_i^* \sigma_i(x) = \chi_{[k]}(x). \tag{F.89}$$

Proof. We assume k to be even. Let $\sum_{j=1}^{k} x_j = k - 2t$, where $t := |\{i : x_i = -1, i \in [k]\}|$. Thus,

$$\sigma_i(x) = \mathrm{ReLU}\left(\frac{2(i+1-t)}{k}\right). \tag{F.90}$$

We choose

$$\begin{aligned} a_i^* &= (-1)^i 2k \qquad \forall i \in \{0,1,...,k-2\}, &(F.91)\\ a_i^* &= (-1)^i \frac{3}{2}k \qquad i = k-1, &(F.92)\\ a_i^* &= (-1)^i \frac{1}{2}k \qquad i = k. &(F.93)\end{aligned}$$

One can check that with these a_i^* the statement holds. □

Lemma 32. *Let* $f^*(x) = \sum_{i=0}^{k} a_i^* \sigma_i(x)$ *and let* $\hat{f}(x) = \sum_{i=0}^{k} a_i^* \hat{\sigma}_i(x)$*, with* $\sigma_i(x)$ *defined above and* $\hat{\sigma}_i(x) := \mathrm{ReLU}(\sum_{j=1}^{d} w_{i,j}^1 x_j + b_i^1)$*. Then, with probability* $1-3\delta$ *for all* x*,*

$$(1 - f(x) f^*(x))_+ \le \frac{32k^2\zeta N}{\mu^{k-1}-\mu^{k+1}} \cdot \left(1+\frac{d-k}{k}\right), \tag{F.94}$$

where $(z)_+ := \max\{0, z\}$.

Proof.

$$\begin{aligned} |f^*(x) - \hat{f}(x)| &= \left| \sum_i a_i^*(\sigma_i(x) - \hat{\sigma}_i(x) \right| && \text{(F.95)} \\ &\leq k \|a^*\|_\infty \sup_i |\sigma_i(x) - \hat{\sigma}_i(x)| && \text{(F.96)} \\ &\leq \frac{32k^2\zeta N}{\mu^{k-1} - \mu^{k+1}} \cdot \left(1 + \frac{d-k}{k}\right). && \text{(F.97)} \end{aligned}$$

Thus,

$$\begin{aligned} (1 - f(x)f^*(x))_+ &\leq |1 - f(x)f^*(x)| && \text{(F.98)} \\ &= |f^{*^2}(x) - f(x)f^*(x)| && \text{(F.99)} \\ &= |f^*(x)| \cdot |f^*(x) - f(x)| \leq \frac{32k^2\zeta N}{\mu^{k-1} - \mu^{k+1}} \cdot \left(1 + \frac{d-k}{k}\right). && \text{(F.100)} \end{aligned}$$

□

F.2.3 Second step: Convergence

We apply Theorem 17 with $\mathscr{L}(a) := \mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[L_{\mathrm{cov}}((a, b^1, w^1), \chi_{[k]}, x)\right]$, $\rho = 2\sqrt{N}$, $\mathscr{B} = 2k\sqrt{k}$. We get that if

1. $N \geq (k+1)\log(\frac{k+1}{\delta})$;
2. $\zeta \leq \frac{\epsilon(\mu^{k-1} - \mu^{k+1})}{64k^2N} \cdot \left(1 + \frac{d-k}{k}\right)^{-1}$;
3. $B \geq (2\zeta^2)^{-1}\log(\frac{dN}{\delta})$;
4. $T \geq \frac{64k^3N}{\epsilon^2}$.

then with probability $1 - 3\delta$ over the initialization,

$$\mathbb{E}_{x\sim \mathrm{Rad}(1/2)^{\otimes d}}\left[\min_{t\in[T]} L_{\mathrm{cov}}\left(\chi_{[k]}, \theta^t, x, \mathrm{Rad}(1/2)^{\otimes d}\right)\right] \leq \epsilon. \tag{F.101}$$

Remark 21. *We remark that if* $\mu = \theta_{d,k}(1)$*, then* ζ *decreases exponentially fast in* k*, and as a consequence the batch size and the computational cost grow exponentially in* k*. If however we choose* $\mu = 1 - 1/k$*, then we get* $\zeta = 1/\mathrm{poly}(k)$ *and, as a consequence, the batch size and the computational cost grow polynomially in* k*.*

F.3 Proof of Theorem 14

Let us consider $r = 2$. The case of general r follows easily. Let us state the following Lemma.

Lemma 33. *Let $x \sim \mathrm{Rad}(p)^{\otimes d}$ and let $H(x) := \sum_{i=1}^d \mathbb{1}(x_i = 1)$ be the Hamming weight of $H(x)$. Assume $\epsilon < p < \epsilon'$ for some $\epsilon, \epsilon' \in [0, 1/2]$, then,*

$$\mathbb{P}_{x\sim\mathrm{Rad}(p)^{\otimes d}}\left(H(x) \ge \epsilon' d\right) \le 2\exp\left(-(\epsilon' - p)^2 d\right); \tag{F.102}$$

$$\mathbb{P}_{x\sim\mathrm{Rad}(p)^{\otimes d}}\left(H(x) \le \epsilon d\right) \le 2\exp\left(-(p - \epsilon)^2 d\right). \tag{F.103}$$

Proof of Lemma 33. We apply Hoeffding's inequality with $\mathbb{E}[H(x)] = pd$ and $\sum_{i=1}^d x_i = d - 2H(x)$:

$$\mathbb{P}(|H(x) - pd| \ge td) \le 2\exp\left(-(t-p)^2 d\right). \tag{F.104}$$

□

Take ϵ such that $|p_1 - \frac{1}{2}| > \epsilon$, and consider the following algorithm:

1. Take a fully connected neural network $\mathrm{NN}(x;\psi)$ with the same architecture as $\mathrm{NN}(x;\theta)$ and with initialization $\psi^0 = \theta^0$;
2. Train $\mathrm{NN}(x;\psi)$ on data $(x, \chi_S(x))$, with $x \sim \mathrm{Rad}(p_1)^{\otimes d}$, for T_1 epochs;
3. Train $\mathrm{NN}(x;\psi)$ with initialization ψ^{T_1} on data $(x, \chi_V(x))$, with $x \sim \mathrm{Rad}(1/2)^{\otimes d}$, for $T - T_1$ epochs.

The result holds by the following two Lemmas.

Lemma 34. $\mathrm{TV}(\theta^T;\psi^T) \le \frac{AT\sqrt{P}}{\tau}\cdot\exp(-d\delta^2)$*, where $\delta = \min\{|\epsilon - p_1|, |1/2 - \epsilon|\}$ and* TV *denotes the total variation distance between the law of θ^T and ψ^T.*

Proof of Lemma 34. Clearly, $\mathrm{TV}(\theta^0;\psi^0) = 0$. Then, using subadditivity of TV

$$\mathrm{TV}(\theta^T;\psi^T) \le \sum_{t=1}^T \mathrm{TV}(\theta^t;\psi^t|\{Z^i\}_{i\le t-2}) \tag{F.105}$$

$$= \sum_{t=1}^T \mathrm{TV}(\gamma(g_{\theta^{t-1}} + Z^{t-1});\gamma(g_{\psi^{t-1}} + Z^{t-1})|\{Z^i\}_{i\le t-1}), \tag{F.106}$$

where $g_{\theta^{t-1}}, g_{\psi^{t-1}}$ denote the population gradients in θ^{t-1} and ψ^{t-1}, respectively. Then, recalling that the Z^{t-1} are Gaussians, we get

$$\begin{aligned}
\mathrm{TV}(\theta^T;\psi^T) &\overset{(a)}{\leq} \sum_{t=1}^{T} \frac{1}{2\tau\gamma} \|\gamma g_{\theta^{t-1}} - \gamma g_{\psi^{t-1}}\|_2 && \text{(F.107)}\\
&\overset{(b)}{\leq} \sum_{t=1}^{T_{r-1}} \frac{1}{2\tau\gamma} \cdot 2\sqrt{P}A\gamma \cdot \mathbb{P}(H(x) \geq \epsilon d) + \sum_{t=T_{r-1}}^{T} \frac{1}{2\tau\gamma} \cdot 2\sqrt{P}A\gamma \cdot \mathbb{P}(H(x) \leq \epsilon d) && \text{(F.108)}\\
&\overset{(c)}{\leq} \frac{AT\sqrt{P}}{\tau} \exp(-d\delta^2). && \text{(F.109)}
\end{aligned}$$

In (a) we applied the formula for the TV between Gaussian variables with same variance. In (b) we used that each gradient is in $[-A, A]$ and that during the first part of training, for all x with $H(x) < \epsilon$, the two gradients are the same, and, similarly, in the second part of training, for all x with $H(x) > \epsilon d$, the two gradients are the same. In (c) we applied Lemma 33. □

We apply Theorem 3 from Abbe and Sandon, 2020a, which we restate here for completeness.

Theorem 19 (Theorem 3 in Abbe and Sandon, 2020a)**.** *Let $\mathscr{P}_k$ be the set of k-parities over d bits. Noisy-GD on any neural network* $\mathrm{NN}(x; w)$ *of size P_w and any initialization, after T steps of training on samples drawn from the uniform distribution, outputs a network such that*

$$\frac{1}{|\mathscr{P}_k|} \sum_{f \in \mathscr{P}_k} \left| \mathbb{E}_{x \sim \mathrm{Rad}(1/2)^{\otimes d}} \left[\mathrm{NN}(x; w^T) \cdot f(x) \right] \right| \leq \frac{T\sqrt{P_w}A}{\tau} \cdot \binom{d}{k}^{-1/2}. \tag{F.110}$$

In our case, this implies:

$$\mathbb{E}_V \left| \mathbb{E}_{x \sim \mathrm{Rad}(1/2)^{\otimes d}} \left[\mathrm{NN}(x; \psi^T) \cdot \chi_V(x) \right] \right| \leq \frac{(T - T_{r-1})\sqrt{P}A}{\tau} \cdot \binom{d}{k_V}^{-1/2}, \tag{F.111}$$

where by $\mathbb{E}_V$ we denote the expectation over set V sampled uniformly at random from all subsets of $[d]$ of cardinality k_V. By Lemma 33, this further implies:

$$\mathbb{E}_V \left| \mathbb{E}_{x \sim \mathrm{Rad}(1/2)^{\otimes d}} \left[\mathrm{NN}(x; \psi^T) \cdot G_{S,V,\epsilon}(x) \right] \right| \leq \frac{(T - T_{r-1})\sqrt{P}A}{\tau} \cdot \binom{d}{k_V}^{-1/2} + \exp(-d\delta^2). \tag{F.112}$$

To conclude our proof, note that:

$$\begin{aligned}
&\mathbb{E}_V \left| \mathbb{E}_x \left[\mathrm{NN}(x, \psi^T) \cdot G_{S,V,\epsilon}(x) \right] \right| && \text{(F.113)}\\
&= \mathbb{E}_V \left[\left| \mathbb{E}_x \left[\mathrm{NN}(x, \psi^T) \cdot G_{S,V,\epsilon}(x) \right] \right| \mid \left| S \cap V \right| = 0 \right] \cdot \mathbb{P}_V(|S \cap V| = 0) && \text{(F.114)}\\
&\quad + \mathbb{E}_V \left[\left| \mathbb{E}_x \left[\mathrm{NN}(x, \psi^T) \cdot G_{S,V,\epsilon}(x) \right] \right| \mid \left| S \cap V \right| > 0 \right] \cdot \mathbb{P}_V(|S \cap V| > 0). && \text{(F.115)}
\end{aligned}$$

One can check that,

$$\mathbb{P}_V(|S \cap V| > 0) = 1 - \frac{2k_S k_V}{d} + O(d^{-2}). \tag{F.116}$$

Moreover, since both the initialization and noisy-GD on fully connected networks are invariant to permutation of the input neurons, for any V such that $|S \cap V| = 0$, the algorithm achieves the same correlation. Thus, applying Lemma 34:

$$\begin{aligned}&\left|\mathbb{E}_x\left[\mathrm{NN}(x;\theta^T)\cdot G_{S,V,\epsilon}(x)\right]\right| \\ &\qquad \le \frac{(T-T_{r-1})\sqrt{P}A}{\tau}\cdot\binom{d}{k_V}^{-1/2} + \frac{2AT\sqrt{P}}{\tau}\exp(-d\delta^2) + \frac{2k_S k_V}{d} + O(d^{-2}).\end{aligned}$$

The argument for general r holds by taking ϵ such that $|p_l - \frac{1}{2}| > \epsilon$ for all $l \in [r-1]$ and by replacing step 2 of the algorithm above with the following:

2. Train $\mathrm{NN}(x;\psi)$ on data $(x, \chi_S(x))$ using a $(r-1)$-CL$((T_1,...,T_{r-1}),(p_1,...,p_{r-1}))$ strategy.

F.4 Proof of Corollary 6

We use the same proof strategy of Theorem 14: specifically, we use the same algorithm for training a network $\mathrm{NN}(x;\psi)$ with the same architecture as $\mathrm{NN}(x;\theta)$, so that Lemma 34 still holds. We import Theorem 3 from Abbe and Sandon, 2020a in the following form.

Theorem 20 (Theorem 3 in Abbe and Sandon, 2020a)**.** *Let $\mathcal{F}$ be the set of k-parities over set $\{d/2+1,...,d\}$. Noisy-GD on any neural network* $\mathrm{NN}(x;w)$ *of size P_w and any initialization, after T steps of training on samples drawn from the uniform distribution, outputs a network such that*

$$\frac{1}{|\mathcal{F}|}\sum_{f\in\mathcal{F}}\left|\mathbb{E}_{x\sim\mathrm{Rad}(1/2)^{\otimes d}}\left[\mathrm{NN}(x;w^T)\cdot f(x)\right]\right| \le \frac{T\sqrt{P_w}A}{\tau}\cdot\binom{d/2}{k}^{-1/2}. \tag{F.117}$$

Similarly as before, Theorem 20 and Lemma 33 imply:

$$\mathbb{E}_V\left|\mathbb{E}_{x\sim\mathrm{Rad}(1/2)^{\otimes d}}\left[\mathrm{NN}(x;\psi^T)\cdot G_{S,V,\epsilon}(x)\right]\right| \le \frac{(T-T_{r-1})\sqrt{P}A}{\tau}\cdot\binom{d/2}{k_V}^{-1/2} + \exp(-d\delta^2), \tag{F.118}$$

where by $\mathbb{E}_V$ we denote the expectation over set V sampled uniformly at random from all subsets of $\{d/2+1,...,d\}$ of cardinality k_V. Since both the initialization and noisy-GD on fully connected networks are invariant to permutation of the input neurons, for any

$V \subseteq \{d/2+1, ..., d\}$, the algorithm achieves the same correlation. Thus, applying Lemma 34:

$$\left|\mathbb{E}_x\left[\mathrm{NN}(x;\theta^T)\cdot G_{S,V,\epsilon}(x)\right]\right| \le \frac{(T-T_{r-1})\sqrt{P}A}{\tau}\cdot\binom{d/2}{k_V}^{-1/2} + \frac{2AT\sqrt{P}}{\tau}\exp(-d\delta^2). \tag{F.119}$$

Bibliography

Abbe, E., Bandeira, A., & Hall, G. (2016). Exact recovery in the stochastic block model. *Information Theory, IEEE Transactions on*, *62*(1), 471–487. https://doi.org/10.1109/TIT.2015.2490670

Abbe, E. (2017). Community detection and stochastic block models: recent developments.

Abbe, E., Bengio, S., Cornacchia, E., Kleinberg, J., Lotfi, A., Raghu, M., & Zhang, C. (2022). Learning to reason with neural networks: generalization, unseen data and boolean measures. *Accepted at NeurIPS 2022*.

Abbe, E., Bengio, S., Lotfi, A., & Rizk, K. (2023). Generalization on the unseen, logic reasoning and degree curriculum. *arXiv preprint arXiv:2301.13105*.

Abbe, E., Boix-Adsera, E., & Misiakiewicz, T. (2023). Sgd learning on neural networks: leap complexity and saddle-to-saddle dynamics. *arXiv preprint arXiv:2302.11055*.

Abbe, E., & Boix-Adserà, E. (2018). An information-percolation bound for spin synchronization on general graphs.

Abbe, E., & Boix-Adserà, E. (2019). Subadditivity beyond trees and the chi-squared mutual information.

Abbe, E., & Boix-Adserà, E. (2022). On the non-universality of deep learning: quantifying the cost of symmetry. *arXiv preprint arXiv:2208.03113*.

Abbe, E., Boix-Adserà, E., Brennan, M., Bresler, G., & Nagaraj, D. (2021). The staircase property: how hierarchical structure can guide deep learning. *Advances in Neural Information Processing Systems*, *34*.

Abbe, E., Boix-Adserà, E., & Misiakiewicz, T. (2022). The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. *Conference on Learning Theory*, 4782–4887.

Abbe, E., & Cornacchia, E. (2023). Work in preparation.

Abbe, E., Cornacchia, E., Gu, Y., & Polyanskiy, Y. (2021). Stochastic block model entropy and broadcasting on trees with survey. *Conference on Learning Theory*, 1–25.

Abbe, E., Cornacchia, E., Hazla, J., & Marquis, C. (2022). An initial alignment between neural network and target is needed for gradient descent to learn. *International Conference on Machine Learning*, 33–52.

Abbe, E., Cornacchia, E., & Lotfi, A. (2023). Work in preparation.

Abbe, E., Kamath, P., Malach, E., Sandon, C., & Srebro, N. (2021). On the power of differentiable learning versus PAC and SQ learning. *Advances in Neural Information Processing Systems, 34*.

Abbe, E., Massoulié, L., Montanari, A., Sly, A., & Srivastava, N. (2018). Group synchronization on grids. *Mathematical Statistics and Learning*. https://hal.archives-ouvertes.fr/hal-01940467

Abbe, E., & Sandon, C. (2020a). On the universality of deep learning. *Advances in Neural Information Processing Systems, 33*, 20061–20072.

Abbe, E., & Sandon, C. (2020b). *Poly-time universality and limitations of deep learning* [arXiv:2001.02992].

Alaoui, A. E., & Montanari, A. (2019). On the computational tractability of statistical estimation on amenable graphs.

Alekhnovich, M. (2003). More on average case vs approximation complexity. *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, 298–307.

Allen-Zhu, Z., & Li, Y. (2020). *Backward feature correction: How deep learning performs deep learning* [arXiv:2001.04413].

Andoni, A., Panigrahy, R., Valiant, G., & Zhang, L. (2014). Learning polynomials with neural networks. *International conference on machine learning*, 1908–1916.

Arora, S., Cohen, N., Hu, W., & Luo, Y. (2019). Implicit regularization in deep matrix factorization. https://doi.org/10.48550/ARXIV.1905.13655

Arpe, J., & Mossel, E. (2008). Agnostically learning juntas from random walks. *arXiv preprint arXiv:0806.4210*.

Avrahami, J., Kareev, Y., Bogot, Y., Caspi, R., Dunaevsky, S., & Lerner, S. (1997). Teaching by examples: implications for the process of category acquisition. *The Quarterly Journal of Experimental Psychology Section A, 50*(3), 586–606.

Barak, B., Edelman, B. L., Goel, S., Kakade, S., Malach, E., & Zhang, C. (2022). Hidden progress in deep learning: sgd learns parities near the computational limit. *arXiv preprint arXiv:2207.08799*.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. (2010). A theory of learning from different domains. *Machine Learning, 79*, 151–175. http://www.springerlink.com/content/q6qk230685577n52/

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Berthier, R. (2022). Incremental learning in diagonal linear networks. *arXiv preprint arXiv:2208.14673*.

Blum, A., Furst, M., Jackson, J., Kearns, M., Mansour, Y., & Rudich, S. (1994). Weakly learning DNF and characterizing statistical query learning using Fourier analysis. *Symposium on Theory of Computing (STOC)*, 253–262.

Blum, A., Kalai, A., & Wasserman, H. (2003). Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM), 50*(4), 506–519.

Bollobás, B., Janson, S., & Riordan, O. (2007). The phase transition in inhomogeneous random graphs. *Random Structures & Algorithms, 31*(1), 3–122.

Boppana, R. B. (1987). Eigenvalues and graph bisection: an average-case analysis. *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, 280–285.

Bshouty, N. H., Mossel, E., O'Donnell, R., & Servedio, R. A. (2005). Learning dnf from random walks. *Journal of Computer and System Sciences, 71*(3), 250–265.

Bui, T., Chaudhuri, S., Leighton, T., & Sipser, M. (1984). Graph bisection algorithins with good average case behavior. *25th Annual Symposium onFoundations of Computer Science, 1984.*, 181–192.

Campos, D. (2021). Curriculum learning for language modeling. *arXiv preprint arXiv:2108.02170*.

Chen, S., Owusu, S., & Zhou, L. (2013). Social network based recommendation systems: a short survey. *2013 international conference on social computing*, 882–885.

Chizat, L., & Bach, F. (2020). Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. https://doi.org/10.48550/ARXIV.2002.04486

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. *Twenty-second international joint conference on artificial intelligence*.

Coja-Oghlan, A., Krzakala, F., Perkins, W., & Zdeborova, L. (2017). Information-theoretic thresholds from the cavity method. *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 146–157. https://doi.org/10.1145/3055399.3055420

Conditional random fields, planted constraint satisfaction, and entropy concentration. (2015). *Theory of Computing, 11*(17), 413–443. https://doi.org/10.4086/toc.2015.v011a017

Cornacchia, E., & Mossel, E. (2023). A mathematical model for curriculum learning for k-parities.

Daniely, A., & Malach, E. (2020). Learning parities with neural networks. *Advances in Neural Information Processing Systems, 33*, 20356–20365.

Deshpande, Y., Abbe, E., & Montanari, A. (2016). Asymptotic mutual information for the balanced binary stochastic block model. *Information and Inference: A Journal of the IMA, 6*(2), 125–170. https://doi.org/10.1093/imaiai/iaw017

Dokholyan, N. V., Li, L., Ding, F., & Shakhnovich, E. I. (2002). Topological determinants of protein folding. *Proceedings of the National Academy of Sciences, 99*(13), 8637–8641.

Dong, Q., Gong, S., & Zhu, X. (2017). Multi-task curriculum transfer deep learning of clothing attributes. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 520–529.

Eames, K. T., & Read, J. M. (2008). Networks in epidemiology. *Bio-Inspired Computing and Communication: First Workshop on Bio-Inspired Design of Networks, BIOWIRE 2007 Cambridge, UK, April 2-5, 2007 Revised Selected Papers*, 79–90.

Elio, R., & Anderson, J. R. (1984). The effects of information order and learning mode on schema abstraction. *Memory & cognition, 12*(1), 20–30.

Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: a survey. *The Journal of Machine Learning Research, 20*(1), 1997–2017.

Evans, W., Kenyon, C., Peres, Y., & Schulman, L. J. (2000). Broadcasting on trees and the ising model. *Ann. Appl. Probab.*, *10*(2), 410–433. https://doi.org/10.1214/aoap/1019487349

Even, M., Pesme, S., Gunasekar, S., & Flammarion, N. (2023). (s) gd over diagonal linear networks: implicit regularisation, large stepsizes and edge of stability. *arXiv preprint arXiv:2302.08982*.

Fukushima, K. (1988). Neocognitron: a hierarchical neural network capable of visual pattern recognition. *Neural networks*, *1*(2), 119–130.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., Airoldi, E. M., et al. (2010). A survey of statistical network models. *Foundations and Trends® in Machine Learning*, *2*(2), 129–233.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., & Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. *international conference on machine learning*, 1311–1320.

Gu, Y., & Polyanskiy, Y. (2023). Uniqueness of bp fixed point for the potts model and applications to community detection. *arXiv preprint arXiv:2303.14688*.

Gunasekar, S., Lee, J., Soudry, D., & Srebro, N. (2018a). Characterizing implicit bias in terms of optimization geometry. https://doi.org/10.48550/ARXIV.1802.08246

Gunasekar, S., Lee, J., Soudry, D., & Srebro, N. (2018b). Implicit bias of gradient descent on linear convolutional networks. https://doi.org/10.48550/ARXIV.1806.00468

Gunasekar, S., Woodworth, B., Bhojanapalli, S., Neyshabur, B., & Srebro, N. (2017). Implicit regularization in matrix factorization. https://doi.org/10.48550/ARXIV.1705.09280

Hebb, D. O. (1949). *The organization of behavior: a neuropsychological theory*. Science editions.

Heimlicher, S., Lelarge, M., & Massoulié, L. (2012). Community detection in the labelled stochastic block model.

Holland, P. W., Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: first steps. *Social networks*, *5*(2), 109–137.

Ivakhnenko, A. G., & Lapa, V. G. (1965). *Cybernetic predicting devices*. CCM Information Corporation.

Jacot, A., Gabriel, F., & Hongler, C. (2018). Neural tangent kernel: convergence and generalization in neural networks. *Advances in neural information processing systems*, *31*.

Ji, Z., & Telgarsky, M. (2019). Characterizing the implicit bias via a primal-dual analysis. https://doi.org/10.48550/ARXIV.1906.04540

Jiang, L., Meng, D., Zhao, Q., Shan, S., & Hauptmann, A. G. (2015). Self-paced curriculum learning. *Twenty-ninth AAAI conference on artificial intelligence*.

Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., & Girshick, R. (2017). Clevr: a diagnostic dataset for compositional language and elementary visual reasoning. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2901–2910.

Kalimeris, D., Kaplun, G., Nakkiran, P., Edelman, B., Yang, T., Barak, B., & Zhang, H. (2019). Sgd on neural networks learns functions of increasing complexity. *Advances in neural information processing systems, 32.*

Kanade, V., Mossel, E., & Schramm, T. (2014). Global and local information in clustering labeled block models.

Kearns, M. (1998). Efficient noise-tolerant learning from statistical queries. *Journal of the ACM, 45*(6), 983–1006.

Kim, Y., Han, S., Choi, S., & Hwang, D. (2014). Inference of dynamic networks using time-course data. *Briefings in bioinformatics, 15*(2), 212–228.

Kim, Y., Son, S.-W., & Jeong, H. (2010). Finding communities in directed networks. *Physical Review E, 81*(1), 016103.

Kumar, R., Raghavan, P., Rajagopalan, S., & Tomkins, A. (1999). Trawling the web for emerging cyber-communities. *Computer networks, 31*(11-16), 1481–1493.

LeCun, Y., Cortes, C., & Burges, C. (2010). Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, 2.*

Liu, R., Lehman, J., Molino, P., Such, F. P., Frank, E., Sergeev, A., & Yosinski, J. (2018). An intriguing failing of convolutional neural networks and the CoordConv solution. *NeurIPS*, 9628–9639. http://dblp.uni-trier.de/db/conf/nips/nips2018.html#LiuLMSFSY18

Lyu, K., & Li, J. (2019). Gradient descent maximizes the margin of homogeneous neural networks. https://doi.org/10.48550/ARXIV.1906.05890

Malach, E., Kamath, P., Abbe, E., & Srebro, N. (2021). Quantifying the benefit of using differentiable learning over tangent kernels. *International Conference on Machine Learning*, 7379–7389.

Malach, E., & Shalev-Shwartz, S. (2019). Is deeper better only when shallow is good? *Advances in Neural Information Processing Systems, 32*, 6429–6438.

Malach, E., & Shalev-Shwartz, S. (2020). Computational separation between convolutional and fully-connected networks. *arXiv preprint arXiv:2010.01369.*

Massoulié, L. (2014). Community detection thresholds and the weak ramanujan property. *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, 694–703.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics, 5*, 115–133.

Mézard, M., & Montanari, A. (2009). *Information, Physics, and Computation.* Oxford University Press.

Miller, J., Taori, R., Raghunathan, A., Sagawa, S., Koh, P. W., Shankar, V., Liang, P., Carmon, Y., & Schmidt, L. (2021). Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. https://doi.org/10.48550/ARXIV.2107.04649

Minsky, M., & Papert, S. (1969). *Perceptrons: an introduction to computational geometry.* MIT Press.

Mitchell, J. C. (1974). Social networks. *Annual review of anthropology, 3*(1), 279–299.

Mok, J., Na, B., Kim, J.-H., Han, D., & Yoon, S. (2022). Demystifying the neural tangent kernel from a practical perspective: can it be trusted for neural architecture search without training? https://doi.org/10.48550/ARXIV.2203.14577

Moroshko, E., Gunasekar, S., Woodworth, B., Lee, J. D., Srebro, N., & Soudry, D. (2020). Implicit bias in deep linear classification: initialization scale vs training accuracy. https://doi.org/10.48550/ARXIV.2007.06738

Morris, M. (1993). Epidemiology and social networks: modeling structured diffusion. *Sociological methods & research*, *22*(1), 99–126.

Mossel, E., Neeman, J., & Sly, A. (2015). Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, *162*(3), 431–461. https://doi.org/10.1007/s00440-014-0576-6

Mossel, E., Neeman, J., & Sly, A. (2016). Belief propagation, robust reconstruction and optimal recovery of block models. *The Annals of Applied Probability*, *26*(4), 2211–2256. https://doi.org/10.1214/15-aap1145

Mossel, E., Neeman, J., & Sly, A. (2018). A proof of the block model threshold conjecture. *Combinatorica*, *38*(3), 665–708. https://doi.org/10.1007/s00493-016-3238-8

Mossel, E., O'Donnell, R., & Servedio, R. A. (2004). Learning functions of k relevant variables. *Journal of Computer and System Sciences*, *69*(3), 421–434.

Mossel, E., & Xu, J. (2015). Local algorithms for block models with side information.

Newman, M. E., & Park, J. (2003). Why social networks are different from other types of networks. *Physical review E*, *68*(3), 036122.

Neyshabur, B., Bhojanapalli, S., McAllester, D., & Srebro, N. (2017). Exploring generalization in deep learning. https://doi.org/10.48550/ARXIV.1706.08947

Neyshabur, B., Tomioka, R., & Srebro, N. (2014). In search of the real inductive bias: on the role of implicit regularization in deep learning. https://doi.org/10.48550/ARXIV.1412.6614

O'Donnell, R. (2014). *Analysis of boolean functions*. Cambridge University Press. https://doi.org/10.1017/CBO9781139814782

Ortiz-Jiménez, G., Moosavi-Dezfooli, S.-M., & Frossard, P. (2021). What can linearized neural networks actually say about generalization? https://doi.org/10.48550/ARXIV.2106.06770

Pesme, S., Pillaud-Vivien, L., & Flammarion, N. (2021). Implicit bias of sgd for diagonal linear networks: a provable benefit of stochasticity. https://doi.org/10.48550/ARXIV.2106.09524

Polyanskiy, Y., & Wu, Y. (2018). Application of information-percolation method to reconstruction problems on graphs.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., & Misra, V. (2022). Grokking: generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2009). *Dataset shift in machine learning*. The MIT Press.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., & Courville, A. (2019). On the spectral bias of neural networks. *International Conference on Machine Learning*, 5301–5310.

Rao, F., & Caflisch, A. (2004). The protein folding network. *Journal of molecular biology*, *342*(1), 299–306.

Razin, N., & Cohen, N. (2020). Implicit regularization in deep learning may not be explainable by norms. https://doi.org/10.48550/ARXIV.2005.06398

Rebeschini, P., & van Handel, R. (2015). Phase transitions in nonlinear filtering. *Electronic Journal of Probability*, *20*(0). https://doi.org/10.1214/ejp.v20-3281

Refinetti, M., Ingrosso, A., & Goldt, S. (2022). Neural networks trained with sgd learn distributions of increasing complexity. *arXiv preprint arXiv:2211.11567*.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, *65*(6), 386.

Ross, B. H., & Kennedy, P. T. (1990). Generalizing from the use of earlier examples in problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *16*(1), 42.

Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., Beery, S., David, E., Stavness, I., Guo, W., Leskovec, J., Saenko, K., Hashimoto, T., Levine, S., Finn, C., & Liang, P. (2021). Extending the wilds benchmark for unsupervised adaptation. https://doi.org/10.48550/ARXIV.2112.05090

Saglietti, L., Mannelli, S. S., & Saxe, A. (2022). An analytical theory of curriculum learning in teacher–student networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2022*(11), 114014.

Sahebi, S., & Cohen, W. W. (2011). Community-based recommendations: a solution to the cold start problem. *Workshop on recommender systems and the social web, RSWEB, 60*.

Sarafianos, N., Giannakopoulos, T., Nikou, C., & Kakadiaris, I. A. (2017). Curriculum learning for multi-task classification of visual attributes. *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2608–2615.

Scott, J. (2002). *Social networks: critical concepts in sociology* (Vol. 4). Taylor & Francis.

Shafto, P., Goodman, N. D., & Griffiths, T. L. (2014). A rational account of pedagogical reasoning: teaching by, and learning from, examples. *Cognitive psychology*, *71*, 55–89.

Shalev-Shwartz, S., et al. (2012). Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, *4*(2), 107–194.

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: from theory to algorithms*. Cambridge university press.

Shalev-Shwartz, S., & Malach, E. (2021). Computational separation between convolutional and fully-connected networks. *International Conference on Learning Representations (ICLR)*.

Shalev-Shwartz, S., Shamir, O., & Shammah, S. (2017). Failures of gradient-based deep learning. *International Conference on Machine Learning*, 3067–3075.

Shen, J., Qu, Y., Zhang, W., & Yu, Y. (2017). Wasserstein distance guided representation learning for domain adaptation. https://doi.org/10.48550/ARXIV.1707.01217

Shi, Y., Larson, M., & Jonker, C. M. (2013). K-component recurrent neural network language models using curriculum learning. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding,* 1–6.

Shi, Y., Larson, M., & Jonker, C. M. (2015). Recurrent neural network language model adaptation with curriculum learning. *Computer Speech & Language, 33*(1), 136–154.

Silverman, R. A., et al. (1972). *Special functions and their applications.* Courier Corporation.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., & Srebro, N. (2017). The implicit bias of gradient descent on separable data. https://doi.org/10.48550/ARXIV.1710.10345

Soviany, P., Ionescu, R. T., Rota, P., & Sebe, N. (2022). Curriculum learning: a survey. *International Journal of Computer Vision*, 1–40.

T. Richardson & R. Urbanke. (2001). An introduction to the analysis of iterative coding systems. In *Codes, systems, and graphical models* (pp. 1–37). Springer.

Tan, Y. S., Agarwal, A., & Yu, B. (2021). A cautionary tale on fitting decision trees to data from additive models: generalization lower bounds. https://doi.org/10.48550/ARXIV.2110.09626

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. (2021). Mlp-mixer: an all-mlp architecture for vision. *Advances in Neural Information Processing Systems, 34.*

Toneva, M., Sordoni, A., Combes, R. T. d., Trischler, A., Bengio, Y., & Gordon, G. J. (2018). An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159.*

Vapnik, V. (1999). *The nature of statistical learning theory.* Springer science & business media.

Vardi, G., Shamir, O., & Srebro, N. (2021). On margin maximization in linear and relu networks. https://doi.org/10.48550/ARXIV.2110.02732

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30.*

Wang, X., Chen, Y., & Zhu, W. (2021). A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Weinshall, D., & Amir, D. (2020). Theory of curriculum learning, with convex loss functions. *Journal of Machine Learning Research, 21*(222), 1–19.

Weinshall, D., Cohen, G., & Amir, D. (2018). Curriculum learning by transfer learning: theory and experiments with deep networks. *International Conference on Machine Learning,* 5238–5246.

Werbos, P. J. (1994). *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting* (Vol. 1). John Wiley & Sons.

Wiles, O., Gowal, S., Stimberg, F., Rebuffi, S.-A., Ktena, I., Dvijotham, K. D., & Cemgil, A. T. (2022). A fine-grained analysis on distribution shift. *International Conference on Learning Representations.* https://openreview.net/forum?id=Dl4LetuLdyK

Winkelbauer, A. (2012). Moments and absolute moments of the normal distribution. *ArXiv, abs/1209.4340*.

Xiong, Y., He, X., Zhao, D., Tian, T., Hong, L., Jiang, T., & Zeng, J. (2021). Modeling multi-species rna modification through multi-task curriculum learning. *Nucleic acids research*, *49*(7), 3719–3734.

Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y., & Ma, Z. (2019). Frequency principle: fourier analysis sheds light on deep neural networks.

Xu, Z.-Q. J., Zhang, Y., & Xiao, Y. (2018). Training behavior of deep neural network in frequency domain. https://doi.org/10.48550/ARXIV.1807.01251

Yu, Q., & Polyanskiy, Y. (2022). Ising model on locally tree-like graphs: uniqueness of solutions to cavity equations. *arXiv preprint arXiv:2211.15242*.

Zanin, M., Cano, P., Buldú, J. M., & Celma, O. (2008). Complex networks in recommendation systems. *Proc. 2nd WSEAS Int. Conf. on Computer Engineering and Applications, World Scientific Advanced Series In Electrical And Computer Engineering. Acapulco, Mexico: World Scientific Advanced Series In Electrical And Computer Engineering*.

Zaremba, W., & Sutskever, I. (2014). Learning to execute. *arXiv preprint arXiv:1410.4615*.

Zdeborová, L., & Krzakala, F. (2016). Statistical physics of inference: thresholds and algorithms. *Advances in Physics*, *65*(5), 453–552.

Zhang, C., Raghu, M., Kleinberg, J. M., & Bengio, S. (2021). Pointer value retrieval: a new benchmark for understanding the limits of neural network generalization. *ArXiv, abs/2107.12580*.

Elisabetta Cornacchia

elisabetta.cornacchia@epfl.ch
https://sites.google.com/view/elisabetta-cornacchia/home
https://www.linkedin.com/in/elisabetta-cornacchia-56b185165/

Research Interests

Theory of neural networks, learning theory, inference on random graphs.

Education

Sept 2019 – on going	**Ph.D. Mathematics, École Polytechnique Fédérale de Lausanne (EPFL).** Chair of Mathematical Data Science. Thesis supervisor: Prof. Emmanuel Abbé.
Sept 2022 – Dec 2022	**Visiting Ph.D. Student, Massachusetts Institute of Technology (MIT).** Institute for Data, Systems and Society (IDSS)
Sept 2017 – July 2019	**M.Sc. Applied Mathematics, École Polytechnique Fédérale de Lausanne (EPFL).** Specialization in Applied Probabilities and Statistics. Thesis title: *Crowd Polarization under Pairwise Interaction*.
Sept 2014 – July 2017	**B.Sc. Mathematics for Engineering, Polytechnic of Turin.** Thesis title: *Epidemic Dynamics over a Graph*.

Experience

Jan 2019 – on going	**Teaching Assistant, École Polytechnique Fédérale de Lausanne (EPFL).** Bachelor courses: Probabilities and Statistics, Advanced Linear Algebra II. Graduate courses: Combinatorial Statistics, Applied Biostatistics.
Aug 2018 – Jan 2019	**Risk and Middle Office Intern, Mercuria Energy Trading S.A.**, London UK.
Sept 2016 – Feb 2017	**Teaching Assistant, Polytechnic of Turin**. Bachelor courses: Mathematical Analysis I and II.

Research Publications

1 Cornacchia, E., & Mossel, E. (2023). A mathematical model for curriculum learning and its application to learning parities, In ***ICML 2023***. https://arxiv.org/abs/2301.13833

2 Abbe, E., Bengio, S., Cornacchia, E., Kleinberg, J., Lotfi, A., Raghu, M., & Zhang, C. (2022). Learning to reason with neural networks: Generalization, unseen data and boolean measures, In ***NeurIPS 2022***. https://arxiv.org/abs/2205.13647

3 Abbe, E., Cornacchia, E., Hązła, J., & Marquis, C. (2022). An initial alignment between neural network and target is needed for gradient descent to learn, In ***ICML 2022***. https://arxiv.org/abs/2202.12846

4 Cornacchia, E., Mignacco, F., Veiga, R., Gerbelot, C., Loureiro, B., & Zdeborová, L. (2022). Learning curves for the multi-class teacher-student perceptron, In ***Machine Learning: Science and Technology (MLST)***. https://arxiv.org/abs/2203.12094

5 Abbe, E., Cornacchia, E., Gu, Y., & Polyanskiy, Y. (2021). Stochastic block model entropy and broadcasting on trees with survey, In ***COLT 2021, Best Student Paper Award***. https://arxiv.org/abs/2101.12601

6 Cornacchia, E., & Hązła, J. (2020). Intransitive dice tournament is not quasirandom, In *Submitted*. https://arxiv.org/abs/2011.10067

7 Cornacchia, E., Singer, N., & Abbe, E. (2020). Polarization in attraction-repulsion models, In ***ISIT 2020***.
https://arxiv.org/abs/2006.05251

Research Talks

Sept 2022	**2022 Mathematical and Scientific Foundations of Deep Learning Annual Meeting**, Simons Foundation, New York City, US.
June 2022	**Youth in High Dimensions**, ICTP, Trieste, Italy.
	DISMA - Eccellenza seminar, Polytechnic of Turin, Turin, Italy.
May 2022	**MoDL monthly meeting**, online.
Sept 2021	**Workshop in Rigorous Evidence for Information-Computation Trade-offs**, EPFL, Lausanne, Switzerland.
Aug 2021	**Conference in Learning Theory (COLT) 2021**, online and Boulder, Colorado, US.
Feb 2020	**Swiss Winter School on Theoretical Computer Science**, Zinal, Switzerland.

Awards

2021	**Best Student Paper Award**, *COLT 2021*
2020	**PhD Teaching Award**, *Section of Mathematics, EPFL*
2015-2017	**Young Talent Project**, *Polytechnic of Turin and Fondazione CRT bank*
2016	**European Innovation Academy (EIA)**
2015	**Women in Engineering**, *Polytechnic of Turin and Fondazione CRT bank*
	FCA Awards Program, *FCA Group*