Université de Montréal

Dynamics of Learning and Generalization in Neural Networks

par

Mohammad Pezeshki

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Discipline

September 23, 2022

Université de Montréal

Faculté des arts et des sciences

Cette thèse intitulée

Dynamics of Learning and Generalization in Neural Networks

présentée par

Mohammad Pezeshki

a été évaluée par un jury composé des personnes suivantes :

Irina Rish
(président-rapporteur)

Guillaume Lajoie
(directeur de recherche)

Yoshua Bengio
(codirecteur)

Aaron Courville
(membre du jury)

Hanie Sedghi
(examinateur externe)

Pierre-Louis Bellec
(représentant du doyen de la FESP)

Résumé

Les réseaux neuronaux sont remarquablement performants pour une grande variété de tâches d'apprentissage automatique et ont eu un impact profond sur la définition même de l'intelligence artificielle (IA). Cependant, malgré leur rôle important dans l'état actuel de l'IA, il est important de réaliser que nous sommes encore loin d'atteindre une intelligence de niveau humain. Une étape cruciale à l'amélioration de la performance des réseaux neuronaux consiste à faire progresser notre compréhension théorique, qui est en retard par rapport aux développements pratiques. Les dynamiques d'optimisation complexes des réseaux neuronaux, qui résultent d'interactions en haute dimension entre les nombreux paramètres du réseau, constituent un défi majeur pour l'élaboration des fondements théoriques de l'apprentissage profond. Ces dynamiques non triviales donnent lieu à des comportements empiriques déroutants qui, dans certains cas, contrastent fortement avec les prédictions théoriques. L'absence de surapprentissage dans les réseaux sur-paramétrés, leur recours à des corrélations fallacieuses et les courbes de généralisation non monotones font partie des comportements de généralisation des réseaux neuronaux qui laissent perplexe.

Dans cette thèse, notre objectif est d'étudier certains de ces phénomènes perplexes en tant que pièces différentes d'un même casse-tête; un casse-tête dans lequel chaque phénomène sert de signal d'orientation pour développer une meilleure compréhension des réseaux neuronaux. Nous présentons trois articles en vue d'atteindre cet objectif; Le premier article sur *multi-scale feature learning dynamics* étudie les raisons qui sous-tendent la courbe de généralisation à double descente observée dans les réseaux neuronaux modernes. L'une des principales conclusions est que la double descente à travers les époques peut être attribuée à l'apprentissage de traits caractéristiques distincts à différentes échelles : Alors que les représentations faciles/rapides à apprendre sont en sur-apprentissage, les représentations plus complexes/lentes commencent à bien apprendre, ce qui entraîne une deuxième descente de l'erreur sur l'ensemble de test. Le deuxième article sur la *famine de gradient* identifie un phénomène fondamental qui peut entraîner une inclination à l'apprentissage dans les réseaux neuronaux. La famine de gradient se produit lorsqu'un réseau neuronal apprend à minimiser la perte en ne capturant qu'un sous-ensemble des traits caractéristiques pertinents à la classification, malgré la présence d'autres traits caractéristiques informatifs qui ne sont pas découverts. La famine de gradient a des conséquences bénéfiques et néfastes dont nous discutons. Le troisième article sur *les méthodes simples de ré-équilibrage des données*

présente une étude empirique sur le problème de la généralisation à des groupes sous-représentés lorsque les données d'entraînement souffrent de déséquilibres importants. Ce travail porte sur les modèles qui généralisent bien en moyenne mais ne parviennent pas à généraliser à des groupes minoritaires. Notre principale conclusion est que des méthodes simples de ré-équilibrage de données permettent d'atteindre l'état de l'art pour la précision sur les groupes minoritaires, ce qui appelle à une examination plus approfondie des valeurs de référence et des méthodes de recherche sur la généralisation en-dehors du support de la distribution.

Nos résultats permettent de mieux comprendre la mécanique interne des réseaux neuronaux et d'identifier les obstacles à la construction de modèles plus fiables, et ont des implications pratiques quant à l'entraînement des réseaux neuronaux.

Mots-clés: les réseaux de neurones, apprentissage automatique, l'apprentissage en profondeur, apprentissage de la représentation

Abstract

Neural networks perform remarkably well in a wide variety of machine learning tasks and have had a profound impact on the very definition of artificial intelligence (AI). However, despite their significant role in the current state of AI, it is important to realize that we are still far from achieving human-level intelligence. A critical step in further improving neural networks is to advance our theoretical understanding which is in fact lagging behind our practical developments. A key challenge in building theoretical foundations for deep learning is the complex optimization dynamics of neural networks, resulting from the high-dimensional interactions between a large number of network parameters. Such non-trivial dynamics lead to puzzling empirical behaviors that, in some cases, appear in stark contrast with existing theoretical predictions. Lack of overfitting in over-parameterized networks, their reliance on spurious correlations, and double-descent generalization curves are among the perplexing generalization behaviors of neural networks.

In this dissertation, our goal is to study some of these perplexing phenomena as different pieces of the same puzzle. A puzzle in which every phenomenon serves as a guiding signal towards developing a better understanding of neural networks. We present three articles towards this goal; The first article on *multi-scale feature learning dynamics* investigates the reasons underlying the double-descent generalization curve observed in modern neural networks. A central finding is that epoch-wise double descent can be attributed to distinct features being learned at different scales: as fast-learning features overfit, slower-learning features start to fit, resulting in a second descent in test error. The second article on *gradient starvation* identifies a fundamental phenomenon that can result in a learning proclivity in neural networks. Gradient starvation arises when a neural network learns to minimize the loss by capturing only a subset of features relevant for classification, despite the presence of other informative features which fail to be discovered. We discuss how gradient starvation can have both beneficial and adverse consequences on generalization performance. The third article on *simple data balancing methods* conducts an empirical study on the problem of generalization to underrepresented groups when the training data suffers from substantial imbalances. This work looks into models that generalize well on average but fail to generalize to minority groups of examples. Our key finding is that simple data balancing methods already achieve state-of-the-art accuracy on minority groups which calls for closer examination of benchmarks and methods for research in out-of-distribution generalization. These three articles take steps towards bringing insights into the inner mechanics of neural networks, identifying the obstacles in the way of building reliable models, and providing practical suggestions for training neural networks.

Keywords: neural networks, machine learning, deep learning, representation learning

Contents

List of tables

List of figures

List of acronyms and abbreviations

AI	artificial intelligence
BCE	binary cross entropy
CDF	cumulative distribution function
DD	double descent
ERM	empirical risk minimization
gDRO	group distributionally robust optimization
GS	gradient starvation
HP	hyper parameters
IID	independent and identically distributed
IRM	invariant risk minimization
JTT	just train twice

LfF	learning from failure
ML	machine learning
MSE	mean square error
NTK	neural tangent kernel
NTRF	neural tangent random feature
OOD	out of distribution
PGD	projected gradient descent
RWY	reweighting classes
SGD	stochastic gradient descent
SOTA	state of the art
SVD	singular value decomposition

Acknowledgment

I have been very fortunate to have the opportunity to meet and receive support from so many friends, colleagues, mentors, and role models in my research journey.

In 2013, I sent an email to Yoshua Bengio as a bachelor student, sharing my very first academic paper. Yoshua replied in ten minutes and that was when good things started to happen. Yoshua graciously offered me admission for master's at Mila (Lisa at the time), introducing me to the world of neural networks, and supporting me both financially and emotionally. I will always be indebted to him for giving me the chance to be the researcher who I am today.

I would like to deeply thank Guillaume Lajoie for his unwavering guidance and inspiration. His encouragement and advice have been crucial for me. I consider myself extremely lucky to have Guillaume as my Ph.D. advisor. Guillaume has always provided me with valuable advice in research. He had a profound impact on my productivity and as his surname suggests, working with Guillaume has been extremely joyful! Something that I love about Guillaume is that he has always offered me guidance not only on high-level research ideas but also on low-level technical details. I will remain grateful for his support.

I am also truly thankful to Aaron Courville from whom I learned a lot about academic research. Aaron is one of the most talented people I have ever met. He has had a critical role in shaping my attitude towards tackling research problems. Aaron always guided me to ask the right questions and be a critical thinker. I feel privileged to have the opportunity to learn from him.

I would like to thank all my co-authors Amartya Mitra, Sékou-Oumar Kaba, Badr Youbi Idrissi, Martin Arjovsky, David Lopez-Paz, Doina Precup, Aaron Courville, Yoshua Bengio, and Guillaume Lajoie. It has been a privilege to collaborate with so many talented people.

In addition to my collaborators, many others have had an important role in shaping my research and also bringing comfort to my life during my studies. I would like to extend my appreciation to Adam Ibrahim, Ali Askari Hemmat, Aristide Baratin, Ataollah Askari Hemmat, Celine Begin, David Yu-Tung Hui, Emmanuel Bengio, Ethan Caballero, Ezekiel Williams, Florian Bordes, Frédéric Bastien, Gabriel Huang, Gauthier Gidel, Hossein Askari Hemmat, Ioannis Mitliagkas, Irina Rish, Julie Mongeau, Kartik Ahuja, Madhu Advani, Maximilian Puelma Touzel, Mohammad Ahmadpanah, Negar Rostamzadeh, Olexa Bilanuik, Pascal Lamblin, and Remi Tachet des Combes, Rémi Le Priol, Samira Shabanian, Shagun Sodhani, Shibl Mourad, Simon Lacoste-Julien, Simon

Lefrancois, Sina Honari, Thomas George, Zahra Ebrahimi.

I am deeply thankful to Reyhane Askari Hemmat, Faruk Ahmed, Amartya Mitra, Vincent Mai, and Colleen Gillon for providing me with invaluable feedback and assisting me in writing. This thesis would not have been possible without your support.

An important part of my research journey has been the internship opportunities. I would like to thank Hugo Larochelle, Layla El Asri, Doina Precup, Diane Bouchacourt, David Lopez-Paz, Pascal Vincent, Nicolas Ballas, and Mark Ibrahim for trusting me and giving me the chance to experience research in big tech companies.

Most importantly, I would love to thank my family. I am deeply thankful to my parents, Parviz and Effat, for their support, sacrifice, patience, encouragement, and sincere love throughout my entire life. My thanks go to my brother, Ehsan, and my sister, Sara, who have had a critical role in shaping the way I see the world. Finally, I present my heartfelt appreciation to my lovely wife and life companion Reyhane Askari Hemmat. She has always been by my side through thick and thin. Her heart-warming support has always cheered me up when I felt being lost during my Ph.D. and even life.

Chapter 1

Introduction

Over the last decade, a surge in the amount of available data combined with increasingly powerful computational resources has resulted in neural networks achieving significant gains across a wide range of tasks (see e.g. Sejnowski, 2018). Despite their outstanding empirical success, neural networks remain unpredictable in many ways. For example, it is not entirely obvious, how a neural network reacts to a particular alternation in the training algorithm, its regularization, or its architecture. This can be exemplified, by the widely used batch normalization method (Ioffe and Szegedy, 2015), whose reasons for effectiveness are still poorly understood. Another example is the existence of adversarial examples (Goodfellow et al., 2014), examples that are intentionally designed to cause misclassification. It is surprising to observe how drastically a neural network's prediction changes when fed with adversarial examples, whose key effectors are imperceptible to humans.

To combat such unpredictabilities, we need a dependable theoretical understanding of neural networks. Systematic study of neural network's learning dynamics paves the path for robust theoretical guarantees, navigate practitioners towards models with improved reliability, and allow for more directed fine-tuning of them.

Theoretical works on neural networks started more than three decades ago with studies on the capacity and generalization of shallow networks (e.g. Baldi and Hornik, 1989a; Heskes and Kappen, 1993a; Le Cun et al., 1991; Watkin et al., 1993; Seung et al., 1992). After neural networks regained popularity in the 2010s, seminal studies (Saxe et al., 2013b, 2019; Advani and Saxe, 2017b; Lampinen and Ganguli, 2018) investigated the learning dynamics of deep linear neural networks, showing that several intuitions of deep linear models carry over to more complex non-linear networks. These works, for example, provided explanations for the effectiveness of depth observed in practice. More recently, significant progress has been made in studying more general deep networks (e.g. Li et al., 2020; Chen et al., 2020b; Chizat and Bach, 2020; Arora et al., 2019c) as well as incorporating the structure of data in their analysis (e.g. Goldt et al., 2019, 2020; Gerace et al., 2020).

Notwithstanding the significant progress made towards establishing theoretical foundations for neural networks, several learning phenomena, well-known in practice, remain opaque when studied in theory. One such phenomenon is the existence of perplexing non-monotonous patterns in the generalization curves of neural networks (see e.g. Bartlett, 1998; Breiman, 2018; Neyshabur et al., 2014; Zhang et al., 2021). This is unexpected as it is predicted by statistical learning theories that generalization error follows a U-shaped curve: as model complexity (e.g. model size) is increased, the generalization error initially descents and then increases beyond a certain threshold, i.e., overfitting occurs (Vapnik, 1998). In practice, however, neural networks may follow a double descent curve (Spigler et al., 2019; Belkin et al., 2019a): a U-shaped curved followed by a second descent as the model size increases. Nakkiran et al. (2019a) show that double descent is not limited to varying the model size but is also observed as the training time proceeds. Once again, the so-called *epoch-wise double descent* is in apparent contradiction to the classical understanding of over-fitting, where one expects that longer training of a sufficiently large model beyond a certain threshold should result in overfitting (Bengio, 2012).

Another problem of neural networks' dynamics is their reliance on spurious correlations. It has been reported that, in many cases, state-of-the-art (SOTA) neural networks tend to focus on low-level superficial correlations, rather than more abstract and robustly informative features of interest (see e.g. Geirhos et al., 2020). As an example, in a recent study conducted by researchers at Cambridge (Roberts et al., 2021), the authors review more than 300 papers on COVID prediction given CT-Scan images. According to the article, none of the papers were able to generalize from one hospital to another since the models tend to latch on to hospital-specific features, namely relying on the specific characters printed at the corner of the images. Due to such reliances, neural networks are vulnerable under slight distributional shifts between the training and test sets (see e.g. Nagarajan et al., 2020).

The vulnerability of neural networks is particularly revealed when they are tested on out-of-distribution (OoD) test data, where the test distribution differs from the training distribution. While SOTA neural networks generally achieve excellent in-domain generalization performance, small discrepancies between training and testing distributions could cause these neural networks to fail in spectacular ways (Alcorn et al., 2019). Most of the existing learning methods rely on the fundamental assumption that training and test data are identically and independently distributed (IID) (Vapnik, 1998). However, in most practical applications, the IID assumption is violated and distributional shifts are observed between the training and the test sets. Neural networks trained with gradient-based optimization methods, latch onto features in the training distribution that might not be of any use at the test time or could even hurt generalization.

On the quest for developing a scientific understanding of neural networks, throughout this dissertation, our guiding question is:

"What features do neural networks learn, in which order, and how does it affect their generalization?"

To answer this question, the puzzling phenomena and failure cases of neural networks provide us with signals in the direction of developing a better understanding of neural networks. In the remainder of this dissertation, we present three articles aimed at answering our guiding question:

- Chapters 3 and 4 present *"Multi-scale Feature Learning Dynamics: Insights for Double Descent"* (Pezeshki et al., 2021) in which we study a linear teacher-student setup exhibiting epoch-wise double descent similar to that observed in deep neural networks. In this setting, we derive closed-form analytical expressions for the evolution of generalization error over the course of training. We find that double descent can be attributed to distinct features being learned at different scales: as fast-learning features overfit, slower-learning features start to fit, resulting in a second descent in test error.

- Chapters 5 and 6 present *"Gradient Starvation: A Learning Proclivity in Neural Networks"* (Pezeshki et al., 2020) in which we identify and formalize gradient starvation (GS), a fundamental gradient descent phenomenon in neural networks. Gradient Starvation arises when the cross-entropy loss is minimized by capturing only a subset of features relevant for the task, despite the presence of other predictive features that fail to be discovered. This work provides a theoretical explanation for the emergence of such feature imbalance in neural networks. Using tools from dynamical systems, we identify simple properties of learning dynamics that lead to this imbalance and prove that such a situation can be expected given certain statistical structures in training data. Based on our proposed formalism, we develop guarantees for a novel regularization method aimed at decoupling feature learning dynamics, improving accuracy and robustness in cases hindered by gradient starvation.

- Chapters 7 and 8 present *"Simple data balancing achieves competitive worst-group-accuracy"* (Idrissi et al., 2021) in which we empirically study the problem of generalization under distributional shifts. We look into classifiers that generalize well to specific groups of data (good average performance) but fail to generalize to underspecified groups (minority examples). After observing that common worst-group-accuracy datasets suffer from substantial imbalances, we set out to compare state-of-the-art methods to a simple balancing of classes and groups by either subsampling or reweighting data. Our results show that these data balancing baselines achieve state-of-the-art accuracy, begging closer examination of benchmarks and methods for research in OoD generalization.

Before diving into the main articles, we review the necessary background in Chapter 2. Chapters 3 through 8 present the three articles. Lastly, Chapter 9 concludes this dissertation.

Chapter 2

Background

2.1. A primer on neural networks

Inspired by the human brain, artificial neural networks (or simply neural networks) are data processing systems composed of large numbers of processing units. In an analogy to the brain, each of these processing units is called a *neuron*. Usually, lots of artificial neurons are connected to one another in a hierarchical layered structure. In the following, we review neural networks' structure and their mathematical formulation.

2.1.1. Basic components

A neuron is a simple function from one or more inputs to a single output. Consider a set of inputs $\mathbf{x} = \{x_1, x_2, ..., x_d\}$ containing d scalar variables. A set of d scalar weights $\mathbf{w} = \{w_1, w_2, ..., w_d\}$ are assigned to each input in addition to a single scalar bias term b. Formally, an artificial neuron $h(\mathbf{x})$ is defined as,

$$h(\mathbf{x}) = g\Big(\sum_{i=1}^{d} w_i x_i + b\Big), \tag{2.1.1}$$

in which $g(.)$ is a nonlinear function called the *activation function*. Consequently, we refer to the term $\sum_i w_i x_i + b$ as *the pre-activation*. For simplicity, the summation term can be written in vector notation,

$$h(\mathbf{x}) = g(\mathbf{w}^T\mathbf{x} + b). \tag{2.1.2}$$

Neural networks are usually organized in a layer-wise structure. Formally, a multi-layer neural network is defined as,

$$\mathbf{h}^{(k)}(\mathbf{x}) = g(\mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}(\mathbf{x})), \tag{2.1.3}$$

in which, $\mathbf{h}^{(k)}(\mathbf{x})$ is the nonlinear output of k^{th} layer and $\mathbf{h}^{(0)}(\mathbf{x}) = \mathbf{x}$. With $\mathbf{h}^{(k-1)}(\mathbf{x}) \in \mathbb{R}^{D^{k-1}}$ and $\mathbf{h}^{(k)}(\mathbf{x}) \in \mathbb{R}^{D^k}$, we have $\mathbf{W}^{(k)} \in \mathbb{R}^{D^k \times D^{k-1}}$ and $\mathbf{b}^{(k)} \in \mathbb{R}^{D^k}$, denoting the weight matrix and the

bias vector, respectively.

According to the universal approximation theorem (Hornik et al., 1989), multi-layer feedforward networks are capable of approximating any measurable function to any desired degree of accuracy given sufficient number of layers and neurons per each layer. However, finding the appropriate network parameters remains a challenging problem. In the next section, we review methods for training neural networks.

2.1.2. Neural networks optimization

In optimization problems, the *loss function* or *objective function* is a function from a set of variables or values to a scalar value. This scalar value is called the *loss*. Usually the objective of the training procedure is to minimize the loss.

Consider a network with parameters θ with input $\mathbf{x}$ and output $f(\mathbf{x}^{(i)};\theta)$. In supervised learning, the quantity of interest is the generalization loss that measures the performance of a network when tested on new unseen examples. Given a distribution $p(\mathbf{x},\mathbf{y})$, the generalization loss is defined as,

$$\mathcal{L}_G(\theta) := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim p(\mathbf{x},\mathbf{y})}[l(f(\mathbf{x};\theta),\mathbf{y})], \tag{2.1.4}$$

in which, depending on the task, $l(f(\mathbf{x}^{(i)};\theta),\mathbf{y}^{(i)})$ can be mean square error (MSE), negative log-likelihood (nll), 0-1 classification loss, or other scalar functions.

In practice, we do not have access to $p(\mathbf{x},\mathbf{y})$ and hence $\mathcal{L}_G(\theta)$ cannot be optimized directly. Instead, the training loss is optimized. Given N pairs of $(\mathbf{x}^{(i)},\mathbf{y}^{(i)})$, the training loss is defined as,

$$\mathcal{L}_T(\theta) := \frac{1}{N}\sum_i l(f(\mathbf{x}^{(i)};\theta),\mathbf{y}^{(i)}). \tag{2.1.5}$$

The training loss is typically optimized using gradient based optimization methods and specifically, using the gradient descent (GD) algorithm and its variants. GD finds a local minimum of a function by taking small steps in the direction of negative gradient and proportional to its magnitude. GD is an iterative process that at each iteration, the parameters are updated using the following update rule,

$$\theta_{t+1} \leftarrow \theta_t - \eta\nabla_{\theta_t}\mathcal{L}_T(\theta_t), \tag{2.1.6}$$

where η is called the *learning rate*.

Gradient descent is the workhorse underlying neural networks' learning dynamics. In the next sections, we review seminal studies on the learning and generalization dynamics of simple linear neural networks.

2.2. Training dynamics in linear networks

On the dynamics of learning in neural network, numerous works have been introduced since the early days of neural networks. Baldi and Hornik (1989b) attempted to understand gradient descent dynamics by analyzing the critical points of the loss landscape in a simple linear network. Heskes and Kappen (1993b) derived online learning evolution equations with respect to time by converting the discrete system of equations into a continuous system. Despite their interesting findings, the derived equations are highly non-linear in general and do not have a closed-from solution. Among more recent publications, Choromanska et al. (2014), Yosinski et al. (2014), Raghu et al. (2017) developed a variety of approaches to better understand the dynamics and provided practical improvements. Central to this work is Saxe et al. (2013a), for which we summarize the main findings.

Saxe et al. (2013a) explore the dynamics of learning in linear networks, as a proxy to non-linear networks. The authors present exact solutions to learning in deep linear neural networks that provide intuitions for common behaviors of learning - such as existence of long plateaus of the loss function followed by fast transitions to lower loss regions. Central to their work are the singular values of the input-output correlation matrix. Below we review the Singular Value Decomposition (SVD) and then reiterate their theoretical findings in linear networks.

Singular Value Decomposition (SVD) (Forsythe and Moler, 1967) is a powerful tool for matrix analysis as it gives an explicit representation of the range, null space, and rank of a matrix.

Formally, if $A \in \mathbb{R}^{m \times n}$ with $m \geq n$, then a singular value decomposition of A is:

$$A = U\Sigma V^T, \qquad \text{where} \quad \Sigma = \begin{pmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \\ & 0 & \end{pmatrix}, \quad \sigma_1 \geq ... \geq \sigma_n \geq 0, \tag{2.2.1}$$

and $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{n \times n}$ are both unitary matrices. Scalars σ_1 to σ_n are called singular values and the columns of matrices U and V are called left and right singular vectors, respectively. If $\text{rank}(A) = r$, then $\sigma_{r+1} = \sigma_{r+2} = ... = \sigma_n = 0$.

2.2.1. Problem setup

Consider a 3-layer linear neural network with N_i neurons in layer i (See Figure. 1). This network maps inputs $X \in \mathbb{R}^{N_1}$ to outputs $y \in \mathbb{R}^{N_3}$ and is trained on a dataset of P training examples $\{x^\mu, y^\mu\}_{1 \leq \mu \leq P}$. Let W^{21} the weight matrix mapping the first and second layers and W^{32} be the weight matrix mapping the second and third layers. Therefore, the network's total input-output map is $\hat{y} = W_{32}W_{21}X$.

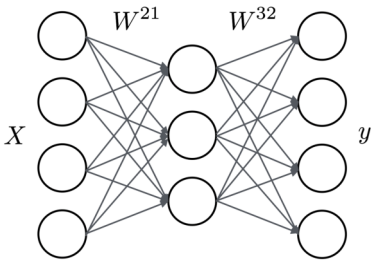


Fig. 1. Depiction of the neural network architecture.

2.2.2. The dynamical system

Consider the loss function associated with this network to be the Mean Square Error (MSE) as following,

$$\mathcal{L}(W_t^{21}, W_t^{32}) = \frac{1}{2}||y - W_t^{32}W_t^{21}X||_2^2. \tag{2.2.2}$$

Minimizing the loss using the gradient descent algorithm leads to the following update rules,

$$W_{t+1}^{21} = W_t^{21} + \eta W_t^{32^T}(yX^T - W_t^{32}W_t^{21}XX^T), \tag{2.2.3}$$

$$W_{t+1}^{32} = W_t^{32} + \eta(yX^T - W_t^{32}W_t^{21}XX^T)W_t^{21^T}. \tag{2.2.4}$$

In the limit of infinitesimal learning rate, $\eta \to 0$, the continuous-time learning dynamics in the parameter space are derived as,

$$\tau\dot{W}^{21} = W^{32^T}(yX^T - W^{32}W^{21}XX^T), \tag{2.2.5}$$

$$\tau\dot{W}^{32} = (yX^T - W^{32}W^{21}XX^T)W^{21^T}, \tag{2.2.6}$$

where τ is a time constant, XX^T is the input correlation matrix, and yX^T is the input-output correlation matrix. To make analysis simpler, we must assume that the input X is whitened and therefore

$$XX^T = I. \tag{2.2.7}$$

This assumption induces that the dynamics are fully governed by yX^T, the input-target covariance matrix. To proceed with the analysis, consider SVD of yX^T,

$$yX^T = USV^T, \tag{2.2.8}$$

in which columns in U and V represent *independent modes of variations* in output and input, respectively. A change of variables such that $\overline{W}^{21} \coloneqq W^{21}V$ and $\overline{W}^{32} \coloneqq U^TW^{32}$ simplifies the dynamics,

$$\tau\dot{\overline{W}}^{21} = \overline{W}^{32^T}(S - \overline{W}^{32}\overline{W}^{21}), \tag{2.2.9}$$

$$\tau\dot{\overline{W}}^{32} = (S - \overline{W}^{32}\overline{W}^{21})\overline{W}^{21^T}. \tag{2.2.10}$$

It is worth noticing that the α^{th} column of $\overline{W}^{21}$ noted as a^α represents all the weights connected to the α^{th} input mode. Likewise, the α^{th} row of $\overline{W}^{32}$ noted as b^{α^T} represents all the weights connected to the α^{th} output mode. We refer to these variables as the connectivity modes (See Figure 2). Derived from Eqs. 2.2.9 and 2.2.10, the dynamics for these variables become

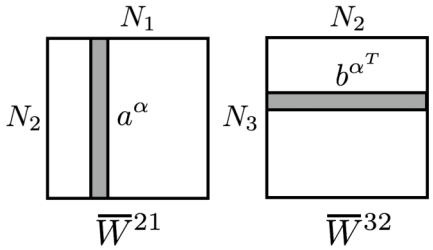


Fig. 2. Depiction of a^α and b^{α^T} as the input and output connectivity modes.

$$\tau \dot{a}^\alpha = (s_\alpha - a^\alpha . \, b^\alpha) b^\alpha - \sum_{\gamma \neq \alpha} b^\gamma (a^\alpha . \, b^\gamma), \tag{2.2.11}$$

$$\tau \dot{b}^\alpha = (s_\alpha - a^\alpha . \, b^\alpha) a^\alpha - \sum_{\gamma \neq \alpha} a^\gamma (b^\alpha . \, a^\gamma). \tag{2.2.12}$$

It can be seen that these dynamics arise from the following loss function,

$$\mathcal{L} \propto \sum_\alpha (s_\alpha - a^\alpha . \, b^\alpha)^2 + \sum_{\alpha \neq \beta} (a^\alpha . \, b^\beta)^2, \tag{2.2.13}$$

which reveals interesting behaviors of the dynamics. We investigate this loss function in more details in Section 2.2.3.

The time course of learning.

Solving Eq. 2.2.11 and Eq. 2.2.12 is not trivial, in general. As a result, we consider a specific initial condition in which,

$$\forall \alpha \neq \beta, \qquad a^\alpha . \, b^\beta = 0. \tag{2.2.14}$$

This initialization requires that for $\alpha = \beta$, a^α and b^β to point along the same direction and only vary in magnitude. Therefore, by defining an orthogonal basis based on vectors r^α, we can project a^α and b^α on r^α such that $a = a^\alpha . \, r^\alpha$ and $b = b^\alpha . \, r^\alpha$. The dynamics of a and b boil down to

$$\tau \dot{a} = b(s - ab), \tag{2.2.15}$$

$$\tau \dot{b} = a(s - ab), \tag{2.2.16}$$

where $s = s_\alpha$. These dynamics correspond to gradient descent on a simplified loss function,

$$\mathcal{L}(a, b) = \frac{1}{2\tau}(s - ab)^2, \tag{2.2.17}$$

which implies that ab gradually approaches s as learning proceeds. Consequently, defining $u \coloneqq ab$ has the following dynamics,

$$\tau \dot{u} = 2u(s - u), \tag{2.2.18}$$

in which u starts at u_0 and ends at u_f. The time required for learning is

$$t = \int_{u_0}^{u_f} \frac{1}{\dot{u}} du \tag{2.2.19}$$

$$= \tau \int_{u_0}^{u_f} \frac{du}{2u(s - u)} \tag{2.2.20}$$

$$= \frac{\tau}{2u} \ln \frac{u_f (s - u_0)}{u_0 (s - u) f)}. \tag{2.2.21}$$

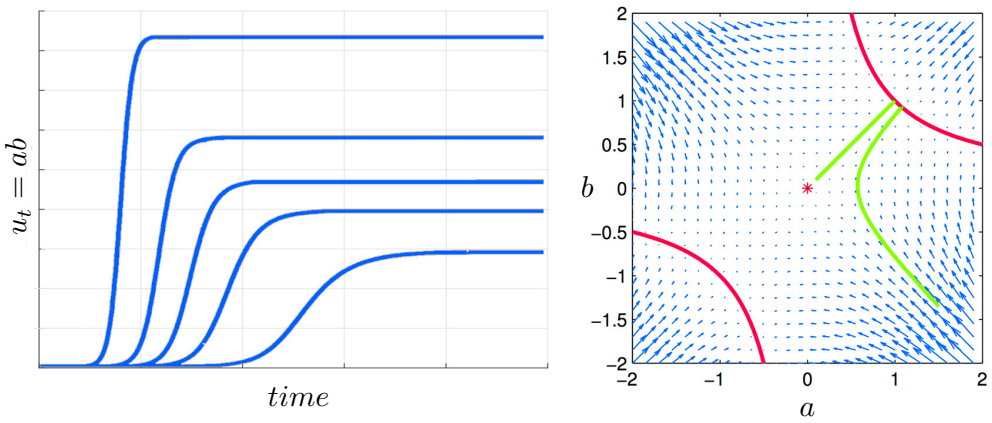


Fig. 3. (left): The sigmoidal-shaped dynamics of mode strength over time. Curves present $u = ab$ for different values of s according to the Eq. 2.2.22 which conforms that modes with larger s converge faster. **(right)**: Vector-filed presentation of phase diagram. According to Eqs. 2.2.15 and 2.2.16, a and b live on hyperbolas with the form $a^2 - b^2 = c$. Two sample trajectories are shown in green. The red curve also presents solutions to $ab = s$. (Figure adapted from Saxe et al. (2013a).)

Inverting this equation leads to the following dynamics for $u(t)$ as function of time,

$$u(t) = \frac{se^{2st/\tau}}{e^{2st/\tau} - 1 + s/u_0}. \tag{2.2.22}$$

2.2.3. Independent Mode Learning: Larger singular values are learned faster

Eq. 2.2.13 exhibits cooperative forces from the first term against competitive forces from the second term. In the first term, the two connectivity modes a^α and b^α cooperate to approach s_α. At the same time, in the second term, the input connectivity mode competes with all other output modes with a different index. This pressures the network to decouple the connectivity modes and create an orthogonal basis.

Minimizing Eq. 2.2.17 leads to dynamics on hyperbolas with constant value of $a^2 - b^2$. A graphical depiction of the phase portrait is shown in Figure 3 (right). Besides, as illustrated in Figure 3 (left), Eq. 2.2.22 shows a sigmoidal-shaped transition from no-learning to full-learning, independent for each mode (input-target singular direction). Interestingly, larger singular values make the sigmoidal pattern rise earlier. In other words, the larger a singular value is, the faster its associated singular direction is learned.

This theoretical finding corresponds to the empirical observation that neural networks exhibit a nonlinear learning behavior, alternating between plateaus and sharp improvement. Saxe et al. (2013a) also presents theoretical results on deep linear neural network. As this dissertation is not focused on *depth*, we leave some of the materials for the interested reader to follow up in the original article.

2.3. Generalization dynamics in linear networks

A principal concept in machine learning is the generalization ability of a learner which measures how well a learner performs when tested on previously unseen data. On the generalization of neural networks, Advani and Saxe (2017a) is among the seminal works that presents an analysis of generalization of a student network in a scenario where a teacher network generates pairs of input/target. Both the student and the teacher are linear networks and the student learns from the teacher by the application of gradient descent. This work analyzes the dynamics of training and generalization by solving the associated dynamical system and provides insights on how the number of examples and the size of a network affect the evolution of training and generalization errors. In the following, we introduce the setup and notations, re-derive the associated dynamical system, and highlight the findings that are most relevant to our topic of interest.

2.3.1. Problem setup

Advani and Saxe (2017a) considers a linear teacher network that generates P pairs of input/target as the training data. The data generation process is also considered noisy by explicitly adding noise to the output of the network. The teacher network follows the following equation,

$$y = \tilde{w}X + \epsilon, \tag{2.3.1}$$

in which $y \in \mathbb{R}^{1\times P}$ is the vector of targets, $X \in \mathbb{R}^{N\times P}$ is the matrix of inputs, $\tilde{w} \in \mathbb{R}^{1\times N}$ is the pre-defined and fixed weight matrix, and $\epsilon \in \mathbb{R}^{1\times P}$ is a noise added to the output of the network. All variables introduced here are drawn as *i.i.d.* from Gaussian distributions with zero means and variances of σ_w^2, σ_ϵ^2, and $\frac{1}{N}$ for $\tilde{w}$, ϵ, and X, respectively. Moreover, the following quantity is introduced to better express the intuitions,

$$\alpha = \frac{P}{N}, \tag{2.3.2}$$

where α distinguishes between over and under-parameterized networks such that an $\alpha < 1$ implies a network with more parameters (N) than the number of training examples (P).

Similarly, the student network follows the following equation,

$$\hat{y} = w(t)X, \tag{2.3.3}$$

in which $w(t)$ is trained to predict outputs for unseen examples of X. $w(t)$, initialized at $w(0)$, is trained using the noisy data generated by the teacher using batch gradient descent. The initialization $w(0)$ is also drawn randomly from a Gaussian distribution with zero mean and a variance of $(\sigma_w^0)^2$.

2.3.2. The dynamical system

The loss function denoted by E_t and associated with training the student network is the Mean Squared Error (MSE),

$$E_t(w(t)) = ||y - \hat{y}||_2^2. \tag{2.3.4}$$

To derive the dynamical equation, we consider a batch gradient descent with infinitesimal learning rate which turns the system into a continuous-time system,

$$\tau \dot{w}(t) = yX^T - wXX^T, \tag{2.3.5}$$

where τ is a time constant, $\Sigma^{XX} \coloneqq XX^T$ is the input correlation matrix, and $\Sigma^{yX} \coloneqq yX^T$ is the input-output correlation matrix.

To solve Eq. 2.3.5, it is useful to decompose XX^T by eigen-decomposition and re-parameterize the system,

$$\Sigma^{XX} = XX^T = V\Lambda V^T. \tag{2.3.6}$$

Considering $X = V\Lambda^{1/2}U^T$, the input-output correlation could be written as follows,

$$\Sigma^{yX} = yX^T = (\bar{w}X + \epsilon)X^T, \tag{2.3.7}$$

$$= \bar{w}XX^T + \epsilon X^T, \tag{2.3.8}$$

$$= \bar{w}V\Lambda V^T + \epsilon U\Lambda^{1/2}V^T, \tag{2.3.9}$$

$$= \bar{w}V\Lambda V^T + \tilde{\epsilon}\Lambda^{1/2}V^T, \tag{2.3.10}$$

where $\tilde{\epsilon} \coloneqq \epsilon U$ is an orthogonal transformation of ϵ and still follows a Gaussian distribution with zero mean and a variance of σ_ϵ^2. Putting everything together, by applying a change of variables, we analyze the vector $z \coloneqq wV$ instead of w in Eq. 2.3.5,

$$\tau \dot{z} = \bar{z}\Lambda + \tilde{\epsilon}\Lambda^{1/2} - z\Lambda, \tag{2.3.11}$$

in which $\bar{z} \coloneqq \bar{w}V$.

Dynamics of each element of z, indexed by i, can be rewritten as,

$$\tau \dot{z}_i = (\bar{z}_i - z_i)\lambda_i + \tilde{\epsilon}_i\sqrt{\lambda_i}. \tag{2.3.12}$$

One apparent result from Eq. 2.3.12 is that learning each mode happens independently of the others. The solution to the above equation is as follows,

$$\bar{z}_i - z_i = (\bar{z}_i - z_i(0))e^{-\frac{\lambda_i t}{\tau}} - \frac{\tilde{\epsilon}_i}{\sqrt{\lambda_i}}(1 - e^{-\frac{\lambda_i t}{\tau}}). \tag{2.3.13}$$

Generalization error dynamics.

Given the learning dynamics of each mode from Eq. 2.3.13, we are interested in the dynamics of the generalization error $E_g(t)$. The generalization error is defined as the expectation of error

marginalized over all potential x and ϵ. Here, we write the generalization error in terms of the modes z as follows[1],

$$E_g(t) = \mathbb{E}_{x,\epsilon}[(\bar{w}X + \epsilon - wX)^2], \tag{2.3.16}$$
$$= \mathbb{E}_x[(\bar{w}X - wX)^2] + \sigma_\epsilon^2, \tag{2.3.17}$$
$$= \mathbb{E}_x[((\bar{z} - z)V^T X)^2] + \sigma_\epsilon^2, \tag{2.3.18}$$
$$= \mathbb{E}_x[(\bar{z} - z)V^T X X^T V(\bar{z} - z)] + \sigma_\epsilon^2, \tag{2.3.19}$$
$$= \mathbb{E}_x[(\bar{z} - z)\Lambda(\bar{z} - z)] + \sigma_\epsilon^2, \tag{2.3.20}$$
$$= \frac{1}{N}\sum_i \lambda_i(\bar{z}_i - z_i)^2 + \sigma_\epsilon^2. \tag{2.3.21}$$

By substituting $(\bar{z}_i - z_i)$ with Eq. 2.3.13 we have,

$$E_g(t) = \frac{1}{N}\sum_i \left[(\sigma_w^2 + (\sigma_w^0)^2)e^{-\frac{2\lambda_i t}{\tau}} + \frac{\sigma_\epsilon^2}{\lambda_i}(1 - e^{-\frac{\lambda_i t}{\tau}})^2\right]\lambda_i + \sigma_\epsilon^2. \tag{2.3.22}$$

A detailed interpretation of Eq. 2.3.22 is provided as we conclude this section.

Training error dynamics.

Unlike the generalization error, for training error there is no expectation as batch gradient descent will go through the whole dataset multiple times. To derive the dynamics, we use the same change of variables as those used in the previous section including $X = V\Lambda^{1/2}U^T$. If $P > N$, we define $\tilde{U} \coloneqq (U, U^\perp) \in R^{P\times P}$ while if $P \leq N$, we define $\tilde{U} \coloneqq U$. In any case, $\tilde{U}\tilde{U}^T = I$. Therefore, the training error can be re-written in the following way,

$$E_t(t) = ||y - wX||_2^2, \tag{2.3.23}$$
$$= ||\bar{w}X + \epsilon - z\Lambda^{1/2}U^T||_2^2, \tag{2.3.24}$$
$$= ||\epsilon + (\bar{z} - z)\Lambda^{1/2}U^T||_2^2, \tag{2.3.25}$$
$$= ||\epsilon\tilde{U} + (\bar{z} - z)\Lambda^{1/2}U^T\tilde{U}||_2^2, \tag{2.3.26}$$
$$= ||\epsilon U + (\bar{z} - z)\Lambda^{1/2}||_2^2 + ||\epsilon U^\perp||_2^2, \tag{2.3.27}$$
$$= ||\tilde{\epsilon} + (\bar{z} - z)\Lambda^{1/2}||_2^2 + ||\epsilon U^\perp||_2^2. \tag{2.3.28}$$

[1]The derivation of generalization error that is provided here is done by the authors of this dissertation and is not provided in the original paper. Advani and Saxe (2017a) mentions the following derivation,

$$E_g(t) = \frac{1}{N}\sum_i (\bar{z}_i - z_i)^2 + \sigma_\epsilon^2. \tag{2.3.14}$$

However, we have derived a slightly different equation,

$$E_g(t) = \frac{1}{N}\sum_i \lambda_i(\bar{z}_i - z_i)^2 + \sigma_\epsilon^2. \tag{2.3.15}$$

Nevertheless, this discrepancy does not change the presented interpretations significantly.

Substituting Eq. 2.3.13 into the equation above yields the following dynamics,

$$E_t(t) = \sum_i^N \left(\sqrt{\lambda_i}(\bar{z}_i - z_i(0)) + \tilde{\epsilon}_i\right)^2 e^{-\frac{2\lambda_i t}{\tau}} + ||\epsilon U^\perp||_2^2. \tag{2.3.29}$$

2.3.3. The effect of input's covariance and noise on generalization dynamics

As it is evident from Eq. 2.3.22, the generalization error dynamics has the following properties,

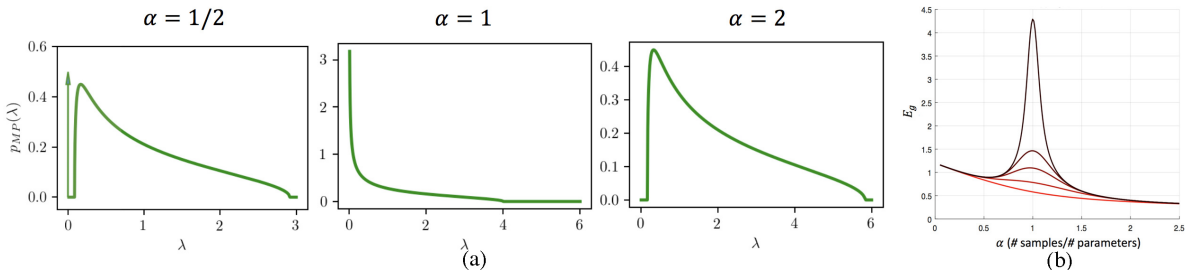
- The term σ_ϵ^2 outside of the brackets presents a time-constant error that bounds the generalization error from below.
- The first term inside of the brackets presents the initialization effect. As t grows, the first term vanishes meaning that more training fades the initialization effect away.
- The second term inside of the brackets starts at zero when $t = 0$. However, as t grows, it approaches to $\frac{\sigma_\epsilon^2}{\lambda_i}$. This term encodes the effects of over-fitting. Interestingly, the smallest but non-zero eigenvalues lead to the most severe over-fitting as the λ_i appears in the denominator.
- Eigenvalues that are exactly zero form a subspace in which no learning occurs. Authors in the original paper call this subspace a *frozen subspace*.
- The smallest eigenvalues not only cause over-fitting but also are slowest to learn. As a result, early stopping is theoretically justified as it prevents these small eigenvalues from causing over-fitting. However, as the learning speed of different modes are not equal, early stopping would cause sub-optimality in learning.

From the properties above, the generalization error dynamics are governed by the λ_i's, eigenvalues of the input correlation/covariance matrix. Consequently, it is desirable to look at the distribution of these eigenvalues in term of model parameters. In the high-dimensional limit, given the ratio α as in Eq. 2.3.2, the eigen-spectrum of XX^T follows a Marchenko-Pasteur distribution (Marchenko and Pastur, 1967),

$$\rho^{\text{MP}}(\lambda) = \frac{1}{2\pi}\frac{\sqrt{(\alpha + 1 + 2\sqrt{\alpha} - \lambda)(\lambda - \alpha - 1 + 2\sqrt{\alpha})}}{\lambda} + 1_{\alpha<1}(1-\alpha)\delta(\lambda), \tag{2.3.30}$$

whose density is depicted in Figure. 4 (a). As discussed, small but non-zero eigenvalues are not desirable for learning. As a result, it is highly preferred to have a large eigen-gap between zero and the smallest eigenvalues which corresponds to an α away from 1. The effect of α is shown in Figure. 4 (b).

Considering the training error dynamics from Eq. 2.3.29, it is evident that the training error always decreases at t grows. Besides, the second term, $||\epsilon U^\perp||_2^2$, bounds the training error from below. However, if $P \leq N$, this term is zero meaning that the network can get zero training error by memorizing every single data-point. It is also interesting to note that stronger λ_i's are faster to learn.



(a) (b)

Fig. 4. **(a)**: Different densities of Marchenko-Pasteur distribution for different values of α. **(b)**: The generalization error as a function of α. Different shades of red presents the amount of training: darker means more training while lighter means less training. In the case of $\alpha = 1$, the number of parameters is equal to the number of data-points. In such a case, there is a large number of small but non-zero eigenvalues that can lead to severe over-fitting. (Figure adapted from Advani and Saxe (2017a))

2.4. The way forward

In this section, we reviewed two seminal works on the dynamics of learning (Sec. 2.2) and generalization (Sec. 2.3) in simple linear neural networks. These and subsequent studies show that various interesting neural network phenomena can already be detected in significantly simpler models. However, as much as simple models are more tractable for analytical studies, they cannot fully characterize the dynamics of complex large-scale neural networks. For example, the results of Sec. 2.2 rely heavily on the assumption that the input covariance matrix is identity (Eq. 2.2.7). This assumption induces *independent mode/feature learning* and enables deriving closed-form analytical solutions. Similarly, in Sec. 2.3, in which the loss function is assumed to be the mean square error (MSE). The gradient of the MSE loss with respect to its inputs is linear, and this linearity is essential for the learning dynamics to be decomposed into independent components.

However, in most practical cases, different features are learned at different rates, implying that the assumption in Eq. 2.2.7 does not hold. Additionally, the majority of modern neural networks are trained using the cross-entropy (CE) loss rather than the MSE loss. Unlike MSE, CE has non-linear derivatives and hence prevents learning dynamics from being decoupled. Relaxation of any of these assumptions would introduce significant challenges but would also enable us to study novel phenomena that result from coupled interactions between the learning of different features. This will be the focus of this dissertation in the following chapters to allow for multi-scale features learning and study the dynamics under the cross-entropy loss.

Chapter 3

Prologue to First Article

3.1. Article Details

Multi-scale Feature Learning Dynamics: Insights for Double Descent. Mohammad Pezeshki, Amartya Mitra, Yoshua Bengio, and Guillaume Lajoie. Proceeding of the thirty-ninth International Conference on Machine Learning (ICML) 2022.

Personal Contributions. Mohammad Pezeshki contributed to the original idea of this work and the writing of the paper. Mohammad Pezeshki initially presented the idea of using the replica method for studying epoch-wise double descent, however, there have been several challenges including the connection to training time and enforcing different speeds of learning for distinct features. Amartya Mitra and Mohammad Pezeshki had several discussions on conquering these challenges. Amartya Mitra significantly contributed to the theory and proofs. Mohammad and Amartya proposed the specific form of the teacher/student and conducted the experiments. Guillaume Lajoie and Yoshua Bengio supervised the project and provided valuable feedback forming the paper. Guillaume Lajoie made several rounds of improvements, provided insights, and significantly contributed to the writing.

3.2. Context

This project summarizes our efforts in explaining a perplexing generalization phenomenon of modern neural networks, namely, the double descent phenomenon. The more commonly studied aspect of this phenomenon corresponds to *model-wise* double descent where the test error exhibits a second descent with increasing model complexity, beyond the classical U-shaped error curve.

There exists a rich literature on studying model-wise double descent and particularly using the replica method of statistical physics. However, the epoch-wise one is much less studied. While the replica method from statistical physics is an effective tool in studying high-dimensional systems, it has the key limitation that it requires various system parameters (including time) to go to infinity. This limitation prevents us from studying the "finite-time dynamics", and thus, transient phenomena such as epoch-wise double descent. As such, we set out to adapt the replica method to the epoch-wise

setting. This is performed through the usage of links between early-stopping and ridge regularization along with the incorporation of anisotropic input features.

3.3. Contributions

This work investigates the origins of the *epoch-wise* double descent. We present a novel teacher-student setup that despite its simplicity exhibits double descent curves similar to those observed in deep neural networks. In this setting, we derive closed-form analytical expressions describing the generalization error in terms of two interpretable scalar macroscopic variables. We provide an explanation for the existence of epoch-wise double descent, suggesting that epoch-wise double descent can be attributed to different features being learned at different time-scales.

Chapter 4

Multi-scale Feature Learning Dynamics: Insights for Double Descent

4.1. Introduction

Classical wisdom in statistical learning theory predicts a trade-off between the generalization ability of a machine learning model and its complexity, with highly complex models less likely to generalize well (Friedman et al., 2001). If the number of parameters measures complexity, deep learning models sometimes go against this prediction (Zhang et al., 2016): deep neural networks trained by stochastic gradient descent exhibit a so-called *double descent* behavior (Spigler et al., 2019; Belkin et al., 2019a) with increasing model parameters. Specifically, with increasing complexity, the generalization error first obeys the classical U-shaped curve consistent with statistical learning theory. However, a second regime emerges as the number of parameters is further increased past a transition threshold where generalization error drops again, hence the "double descent" or more accurately *model-wise double descent*.

Nakkiran et al. (2019a) showed that the phenomenon of double descent is not limited to varying model size and is also observed as a function of training time or epochs. In this case as well, the so-called *epoch-wise double descent* is in apparent contradiction with the classical understanding of overfitting (Vapnik, 1998), where one expects that longer training of a sufficiently large model beyond a certain threshold should result in overfitting. This has important implications for practitioners and raises questions about one of the most widely used regularization method in deep learning (Goodfellow et al., 2016): early stopping. Indeed, while one might expect early stopping to prevent overfitting, it might in fact prevent models from being trained at their fullest potential.

While there has been significant interest, starting from 1990s, to understand the origins of the non-trivial generalization behaviors of neural networks (Opper, 1995; Opper and Kinzel, 1996; Ba et al., 2019; Mei and Montanari, 2019a; d'Ascoli et al., 2020; Gerace et al., 2020), the majority of this previous work has been to understand the *asymptotic* or end-of-training model performance.

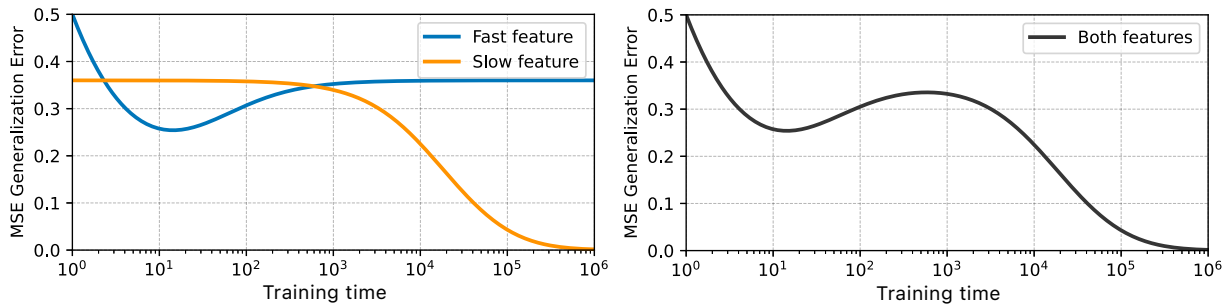


Fig. 1. The generalization error as the training time proceeds. (left): The case where only the fast-learning feature or slow-learning feature are trained. (right): The case with both features. Features that are learned on a faster time-scale are responsible for the classical U-shaped generalization curve, while the second descent can be attributed to the features that are learned at a slower rate.

In recent years though, there has been an interest in studying the *non-asymptotic* (finite training) performance (e.g. Saxe et al., 2013b; Advani and Saxe, 2017b; Kalimeris et al., 2019; Pezeshki et al., 2020; Stephenson and Lee, 2021). Among the limited work studying the particular epoch-wise double descent, Nakkiran et al. (2019a) introduces the notion of *effective model complexity* and hypothesizes that it increases with training time and hence unifies both model-wise and epoch-wise double descent phenomena. Heckel and Yilmaz (2020) also study the dynamics of evolution of single and two layer networks and show that the superposition of two bias/variance trade-off curves with different minima leads to a double descent.

In this work, we build on Bös et al. (1993); Bös (1998); Advani and Saxe (2017b); Mei and Montanari (2019a) which analyze *model-wise* double descent through the lens of linear models, **to probe the origins of *epoch-wise* double descent**. In particular,

- We introduce a linear teacher-student model with features of different strengths. Despite its simplicity, such a model exhibits the epoch-wise double descent of the generalization error under gradient-based training. (Section 4.2.1)
- In the high-dimensional limit (of number of parameters and sample size), we derive the dynamics of a pair of low-dimensional macroscopic variables, R and Q, describing the generalization behavior of the model. (Eqs. 4.2.6, 4.2.7)
- Consistent with recent findings, we provide an explanation for the existence of epoch-wise double descent, suggesting that epoch-wise double descent can be attributed to different features being learned at different time-scales. (Figure 1 and Eqs. 4.2.12-4.2.14)
- We perform simulation experiments to validate our analytical predictions. Furthermore, we conduct experiments with a ResNet-18 model, to demonstrate qualitative similarity between the generalization behavior of our teacher-student setup and that of the former. (Figures 5, 6)

4.2. Analytical Framework

In this work, we focus on studying the generalization behavior of neural networks under the quintessential gradient-based training scenario, namely (stochastic) gradient descent (SGD/GD). SGD — the de facto optimization algorithm for neural networks — exhibits complex dynamics arising from a large number of parameters (Kunin et al., 2020). While an exact analysis of such dynamics is intractable due to the large number of *microscopic* parameters, it is though possible to capture various aspects of this high-dimensional dynamics in terms of certain low-dimensional comprehensible *macroscopic* entities. This was first demonstrated in a series of seminal papers by Gardner (Gardner, 1988; Gardner and Derrida, 1988, 1989), where the *replica method* of statistical physics was adopted to derive expressions describing the generalization behavior of linear models. In this paper, we employ Gardner's analysis to build upon an established line of work studying linear and generalized linear models (Seung et al., 1992; Kabashima et al., 2009; Krzakala et al., 2012). While most of previous work study the asymptotic ($t \to \infty$) generalization behavior, we adapt these methods to study transient learning dynamics of generalization for finite training time. In the following, we introduce a particular linear teacher-student model and study its generalization performance as a function of training time and regularization strength.

Notation. Scalar variables are denoted in lower case (y), while vectorial entities are represented in boldface ($\boldsymbol{x}$). Lastly, matrices are shown capitalized (F).

4.2.1. A Teacher-Student Setup

Teacher. We study a supervised linear regression problem in which the training labels y, are generated by a noisy linear model (Figure 2),

$$y := y^* + \epsilon, \quad y^* := \boldsymbol{z}^T \boldsymbol{w}, \quad z_i \sim \mathcal{N}(0, \frac{1}{\sqrt{d}}), \tag{4.2.1}$$

where $\boldsymbol{z} \in \mathbb{R}^d$ is the teacher's input and $y^*, y \in \mathbb{R}$ are the teacher's noiseless and noisy outputs, respectively. $\boldsymbol{w} \in \mathbb{R}^d$ represents the (fixed) weights of the teacher and $\epsilon \in \mathbb{R}$ is the label noise. Here, both w_i and ϵ are drawn i.i.d. from Gaussian distributions with zero mean and variances of 1 and σ_ϵ^2, respectively. Additionally, we choose to set $||\boldsymbol{w}|| = 1$, without loss of generality.

Student. A student model is correspondingly chosen to be a similar shallow network with trainable weights $\hat{\boldsymbol{w}} \in \mathbb{R}^d$. The student model is trained on n training pairs $\{(\boldsymbol{x}^\mu, y^\mu)\}_{\mu=1}^n$, with the labels y^μ being generated by the above teacher network and where student's inputs $\boldsymbol{x}^\mu$ correspond to teacher inputs $\boldsymbol{z}^\mu$ multiplied a predefined and fixed **modulation matrix** $\mathrm{F} \in \mathbb{R}^{d \times d}$ that regulates input features' strengths:

$$\hat{y} := \boldsymbol{x}^T \hat{\boldsymbol{w}}, \qquad s.t. \qquad \boldsymbol{x} := \mathrm{F}^T \boldsymbol{z}. \tag{4.2.2}$$

One can perceive $\boldsymbol{z}$ to be the latent factors of variation on which the teacher operates, while $\boldsymbol{x}$ corresponds to the pixels that the student learns from. (See Figure 2)

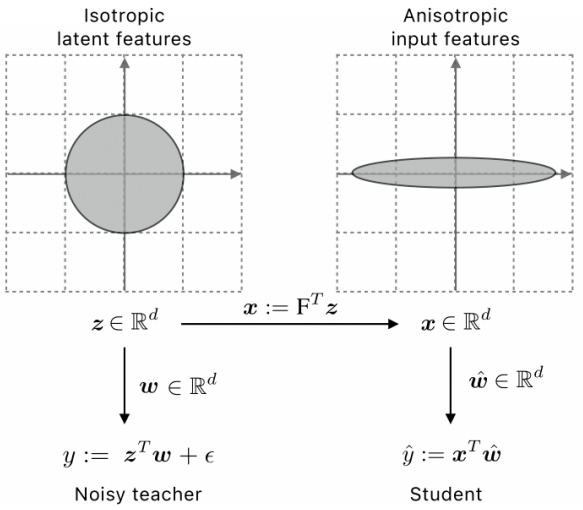


Fig. 2. The teacher/student setup: The teacher is the data generating process that given the latent features in $\boldsymbol{z}$, generates student's input, $\boldsymbol{x}$ and its target, y. Student is trained on pairs of $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ where $\boldsymbol{x} := \mathrm{F}^T \boldsymbol{z}$ follow an anisotropic Gaussian distribution such that the directions with larger/smaller variance are learned faster/slower. The condition number of F determines how much faster some features are learned than the others. One can think of $\boldsymbol{z}$ as the latent factors of variation on which the teacher operates, while $\boldsymbol{x}$ can be thought as the pixels that the student learns from.

Learning paradigm. To train our student network, we use stochastic gradient descent (SGD) on the regularized mean-squared loss, evaluated on the n training examples as,

$$\mathcal{L}_T := \frac{1}{2n} \sum_{\mu=1}^{n} (y^\mu - \hat{y}^\mu)^2 + \frac{\lambda}{2} ||\hat{\boldsymbol{w}}||_2^2, \tag{4.2.3}$$

where $\lambda \in [0, \infty)$ is the regularization coefficient. Optimizing Eq. 6.3.5 with stochastic gradient descent (SGD) yields the typical update rule,

$$\hat{\boldsymbol{w}}_t \leftarrow \hat{\boldsymbol{w}}_{t-1} - \eta \nabla_{\hat{\boldsymbol{w}}} \mathcal{L}_T + \xi, \tag{4.2.4}$$

in which t denotes the training step and η is the learning rate. Following the setup of Kuhn and Bos (1993), $\xi \sim \mathcal{N}(0, \frac{2}{\beta})$ approximates the stochasticity noise of the optimization algorithm, with β corresponding to an inverse *temperature parameter*. The shape of the noise is assumed to be Gaussian by virtue of the central limit theorem. See Bottou et al. (1991); Mandt et al. (2017); Wu et al. (2020) for more details on modeling the stochasticity of SGD with Gaussian noise.

Macroscopic variables. The quantity of interest in this work is the average generalization error of the student determined by averaging the student's error over all possible input-target pairs of a **noiseless** teacher, as

$$\mathcal{L}_G := \frac{1}{2} \mathbb{E}_z \Big[(y^* - \hat{y})^2 \Big]. \tag{4.2.5}$$

As shown in Bös et al. (1993), if $n, d \to \infty$ with a constant ratio $\frac{n}{d} < \infty$, Eq. 4.2.5 can be written as a function of two macroscopic scalar variables $R, Q \in \mathbb{R}$,

$$\mathcal{L}_G = \frac{1}{2}(1 + Q - 2R), \tag{4.2.6}$$

where,

$$R := \frac{1}{d}\boldsymbol{w}^T \mathrm{F}\hat{\boldsymbol{w}}, \qquad Q := \frac{1}{d}\hat{\boldsymbol{w}}^T \mathrm{F}^T \mathrm{F}\hat{\boldsymbol{w}}, \tag{4.2.7}$$

(See App. .2.1 for Proof.)

Remark: Both R and Q have clear interpretations; R is the dot-product between the teacher's weights $\boldsymbol{w}$ and the student's *modulated* weights $\mathrm{F}\hat{\boldsymbol{w}}$, hence can be interpreted as the **alignment between the teacher and the student**. Similarly, Q can be interpreted as the **student's modulated norm**. The negative sign of R in Eq. 4.2.6 suggests that the larger R is, the smaller the generalization error gets. At the same time, Q appears with a positive sign suggesting the students with smaller (modulated) norm generalize better.

Note that both R and Q are functions of $\hat{\boldsymbol{w}}$, which itself is a function of training iteration t and the regularization coefficient λ. Therefore, from hereon, we denote the above quantities as $\mathcal{L}_G(t, \lambda)$, $R(t, \lambda)$, and $Q(t, \lambda)$.

4.2.2. Main Results

In this Section, we present our main analytical results, with Section 4.2.3 containing a sketch of our derivations. For brevity, here, we only present the results for $\sigma_\epsilon^2 = \lambda = 0$. See App. .2 for the general case and the detailed proofs.

General matrix F. Let $\mathrm{Z} := [\boldsymbol{z}^\mu]_{\mu=1}^n \in \mathbb{R}^{n \times d}$ and $\mathrm{X} := [\boldsymbol{x}^\mu]_{\mu=1}^n \in \mathbb{R}^{n \times d}$ denote the input matrices for the teacher and student such that $\mathrm{X} := \mathrm{ZF}$. For a general modulation matrix F, the input covariance matrix has the following singular value decomposition (SVD),

$$\mathrm{X}^T\mathrm{X} = \mathrm{F}^T\mathrm{Z}^T\mathrm{ZF} = \mathrm{V}\Lambda\mathrm{V}^T, \tag{4.2.8}$$

with Λ containing the singular values of the student's input covariance matrix. Solving the dynamics of exact gradient descent as in Eq. 4.2.4, we arrive at the following exact analytical expressions for $R(t)$ and $Q(t)$,

$$R(t) = \frac{1}{d}\mathbf{Tr}\left(\mathrm{D}\right), \quad \text{where,} \quad \mathrm{D} := \mathrm{I} - \left[\mathrm{I} - \eta\Lambda\right]^t, \tag{4.2.9}$$

$$Q(t) = \frac{1}{d}\mathbf{Tr}\left(\mathrm{A}^T\mathrm{A}\right), \text{ where, } \mathrm{A} := \mathrm{FVDV}^T\mathrm{F}^{-1}, \tag{4.2.10}$$

in which $\mathbf{Tr}(.)$ is the trace operator. (See App. .2.2 for Proof.)

By plugging Eqs. 4.2.9 and 4.2.10 into Eq. 4.2.6, one obtains an exact expression for $\mathcal{L}_G(t)$. Unfortunately, Eqs. 4.2.9 and 4.2.10 are not straightforward to treat generally, and require the numerical evaluation of the singular values in Λ. Nevertheless, with some simple but informative

assumptions on the modulation matrix F's structure, one can derive approximate solutions, as we now demonstrate.

Bipartite matrix F. We now study a case where F obeys the following Assumption.

Assumption 4.2.1. The modulation matrix, F, under a SVD, $\mathrm{F} := \mathrm{U}\Sigma\mathrm{V}^T$ has two sets of singular values such that the first p singular values are equal to σ_1 and the remaining $d-p$ singular values are equal to σ_2. We let the condition number of F to be denoted by $\kappa := \frac{\sigma_1}{\sigma_2} \geq 1$.

By employing the replica method of statistical physics (Gardner, 1988; Gardner and Derrida, 1988) and approximation of gradient descent dynamics with ridge regression, we derive closed-form expressions for $R(t)$ and $Q(t)$. To present the results, we first define the following auxiliary variables,

$$\alpha_1 := \frac{n}{p}, \qquad \alpha_2 := \frac{n}{d-p}, \tag{4.2.11}$$

$$\tilde{\lambda}_1 := \frac{d}{p}\underbrace{\frac{1}{\eta\sigma_1^2 t}}_{\text{time scaled by } \sigma_1^2}, \qquad \tilde{\lambda}_2 := \frac{d}{d-p}\underbrace{\frac{1}{\eta\sigma_2^2 t}}_{\text{time scaled by } \sigma_2^2}, \tag{4.2.12}$$

and also let, for $i \in \{1, 2\}$,

$$a_i = 1 + \frac{2\tilde{\lambda}_i}{(1-\alpha_i-\tilde{\lambda}_i) + \sqrt{(1-\alpha_i-\tilde{\lambda}_i)^2 + 4\tilde{\lambda}_i}}. \tag{4.2.13}$$

The scalar expression for $R(t)$ is then given by,

$$\begin{aligned} &R(t) = R_1 + R_2, \quad \text{where,} \\ &R_1 := \frac{n}{a_1 d}, \quad \text{and,} \quad R_2 := \frac{n}{a_2 d}. \end{aligned} \tag{4.2.14}$$

Similarly, for $Q(t)$, we have, $Q(t) = Q_1 + Q_2$, where

$$Q_1 := \frac{b_1 b_2 c_2 + b_1 c_1}{1 - b_1 b_2}, \quad \text{and,} \quad Q_2 := \frac{b_1 b_2 c_1 + b_2 c_2}{1 - b_1 b_2}. \tag{4.2.15}$$

with ($i \in \{1, 2\}$),

$$b_i = \frac{\alpha_i}{a_i^2 - \alpha_i}, \quad c_i = 1 - 2R_i - \frac{n}{d}\frac{2 - a_i}{a_i}, \tag{4.2.16}$$

Plugging Eqs. 4.2.14 and 4.2.15 into Eq. 4.2.6, one obtains an (approximate) expression for $\mathcal{L}_G(t)$ as a function of the training time. (See App. .2.3 for Proof.)

Remark: Eq. 4.2.12 indicates that the singular values of F, are directly multiplied by t. That implies that the learning speed of each feature is scaled by the magnitude of its corresponding singular value.

4.2.3. Sketch of derivations

In this Section, we sketch the key steps in the derivation of our main results. For the sake of simplicity, here again we only treat the case where $\sigma_\epsilon = \lambda = 0$. (See App. .2 for the general case and detailed Proofs.)

General matrix F: Exact dynamics. Recall the gradient descent update rule in Eq. 4.2.4. For the linear model defined in Eqs. 4.2.1-4.2.2, learning is governed by the following discrete-time dynamics,

$$\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} - \eta \nabla_{\hat{\boldsymbol{w}}_{t-1}} \mathcal{L}_T, \tag{4.2.17}$$

$$= \hat{\boldsymbol{w}}_{t-1} - \eta \left[-\mathrm{X}^T (y - \mathrm{X}\hat{\boldsymbol{w}}_{t-1}) \right]. \tag{4.2.18}$$

With the assumption that $\hat{\boldsymbol{w}}_{t=0} = \mathbf{0}$, the dynamics admit the following exact closed-form solution,

$$\hat{\boldsymbol{w}}_t = \left(I - \left[I - \eta \mathrm{X}^T \mathrm{X} \right]^t \right) (\mathrm{X}^T \mathrm{X})^{-1} \mathrm{X}^T y := \tilde{\boldsymbol{w}}(t). \tag{4.2.19}$$

With a SVD on $\mathrm{X}^T\mathrm{X}$, Eqs. 4.2.9-4.2.10 can then be obtained by substituting $\hat{\boldsymbol{w}}_t$ in Eq. 4.2.7. As a remark, note that one can recover the results of Advani and Saxe (2017b) by setting $\mathrm{F} = \mathbb{I}$. In that case, the eigenvalues of $\mathrm{X}^T\mathrm{X}$ follow a Marchenko–Pastur distribution (Marchenko and Pastur, 1967).

Bipartite matrix F: Approximate dynamics. To employ the replica method, we first invoke the results in Eq. 9 of Solla (1995) and Kuhn and Bos (1993) which state that the equilibrium distribution of weights $\hat{\boldsymbol{w}}$ trained via SGD on a loss $\mathcal{L}(\hat{\boldsymbol{w}})$, follow the Gibbs-Boltzmann distribution, such that,

$$P(\hat{\boldsymbol{w}}) = \frac{1}{Z_\beta} e^{-\beta \mathcal{L}(\hat{\boldsymbol{w}})}, \tag{4.2.20}$$

in which $Z_\beta = \int \mathrm{d}\hat{\boldsymbol{w}} \exp(-\beta \mathcal{L}(\hat{\boldsymbol{w}}))$ is the partition function and β is called the *inverse temperature* and is inversely proportional to the stochasticity of SGD (see Eq. 4.2.4). Such distribution is a standard choice in statistical mechanics (see page 53 of Engel and Van den Broeck (2001)). Intuitively, for small β, the distribution of $P(\hat{\boldsymbol{w}})$ is almost uniform, while as $\beta \to \infty$, $P(\hat{\boldsymbol{w}})$ becomes more concentrated around the minimum of the loss $\mathcal{L}(\hat{\boldsymbol{w}})$.

It is important to highlight that Eq. 4.2.20 describes the *equilibrium* distribution of the student network's weights, i.e., at the end of training ($t \to \infty$). However, we are interested in studying the trajectory of student's weights *during* the course of training, i.e., for finite t. To this end, we employ the connection between (continuous-time) SGD and L_2 regularization, as first quantified in Ali et al. (2019, 2020). Specifically, it states that the MSE loss of a linear regression model under stochastic gradient flow at time t is bounded from above by the end-of-training loss in the presence of ridge regression with an L_2 regularization coefficient $\lambda = 1/\eta t$. We note that while there is no guarantee that this bound is tight in general, we do observe that it matches the behavior of a wide range of

numerical experiments extremely well (see Section 4.3).

Accordingly, we study the equilibrium distribution of the modified loss $\tilde{\mathcal{L}}(\hat{\boldsymbol{w}}, t)$, such that,

$$P(\hat{\boldsymbol{w}}) = \frac{1}{Z_{\beta,t}} e^{-\beta \tilde{\mathcal{L}}(\hat{\boldsymbol{w}},t)}, \quad \text{and,} \tag{4.2.21}$$

$$\tilde{\mathcal{L}}(\hat{\boldsymbol{w}}, t) := \frac{1}{2n} \sum_{\mu=1}^{n} (y^\mu - \hat{y}^\mu)^2 + \frac{1}{2}\Big(\lambda + \frac{1}{\eta t}\Big)||\hat{\boldsymbol{w}}||_2^2. \tag{4.2.22}$$

See App. .2.4 for proof.

To determine the *typical* generalization performance of students distributed according to $P(\hat{\boldsymbol{w}})$, one proceeds by computing the free-energy of the system as,

$$f := -\frac{1}{\beta d} \mathbb{E}_{w,\boldsymbol{z}}\Big[\ln Z_{\beta,t}\Big]. \tag{4.2.23}$$

Free-energy is a self-averaging property where its *typical/most probable* value coincides with its *average* over proper probability distributions (Engel and Van den Broeck, 2001). Therefore, to determine the typical values of R and Q, we extremize the free-energy w.r.t. those variables.

Due to the logarithm inside the expectation, analytical computation of Eq. 4.2.23 is intractable. However, the replica method (Mézard et al., 1987) allows us to tackle this through the following identity,

$$\mathbb{E}_{w,\boldsymbol{z}}[\ln Z_{\beta,t}] = \lim_{r \to 0} \frac{\mathbb{E}_{w,\boldsymbol{z}}[Z_{\beta,t}^r] - 1}{r}. \tag{4.2.24}$$

Computation of the free-energy via replica method and its subsequent extremization w.r.t R and Q, we arrive at Eqs. 4.2.14 and 4.2.15. See App. .2.3 for more details.

To summarize, using the replica method, we are able to cast the high-dimensional dynamics of SGD into simple scalar equations governing R and Q and, consequently, the generalization error $\mathcal{L}_G$. While our analysis is limited to the specific teacher and student setup, this simple model already exhibits dynamics qualitatively similar to those observed in more complex networks, as we now illustrate.

4.3. Experimental Results

In this Section, we conduct numerical simulations to validate our analytical results and provide clear insights on the macroscopic dynamics of generalization. We also conduct experiments on real-world neural networks showing a close qualitative match between the generalization behavior of neural networks and our teacher-student setup. To ensure reproducibility, we include the complete source code in a `GitHub repository` as well as a `Colab notebook`.

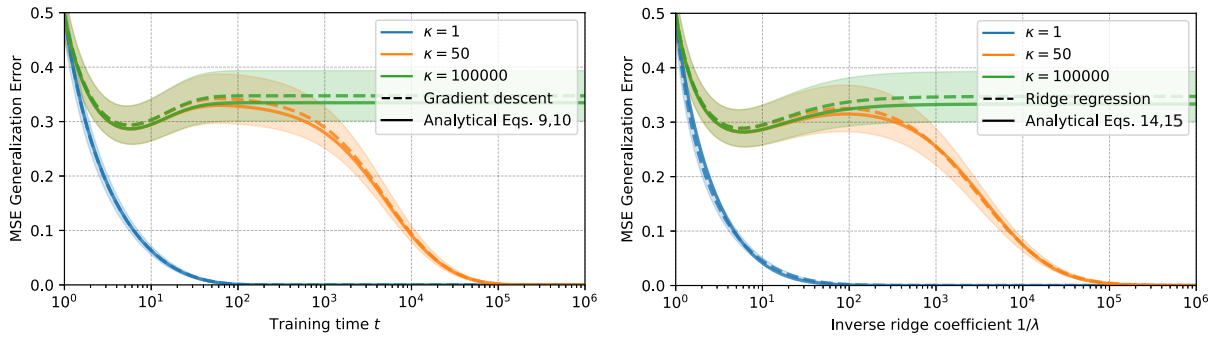


Fig. 3. Left: Analytical results of Eqs. 4.2.9, 4.2.10 compared to gradient descent dynamics. The x-axis denotes the training time t. **Right:** Analytical results of scalar Eqs. 4.2.14, 4.2.15 compared to ridge regression dynamics. The x-axis denotes the inverse ridge (L2) coefficient $1/\lambda$. Analytical results closely match with empirical simulations. Consistent with Ali et al. (2019), ridge regression appears to reasonably approximate gradient descent dynamics. **Analysis:** With $\kappa = 1$, all the features are learned at the same rate (no double descent). $\kappa = 50$ corresponds to the case where a subset of features are learned 50 times faster than the rest and hence epoch-wise double descent is observed. Finally, $\kappa = 100000$ implies that a subset of of features are extremely slow to learn that practically do not get learned (typical overfitting).

4.3.1. Analytical results compared with simulations

Through numerical simulations, we validate our analytical results presented in Section 4.2.2. Figure 3 depicts the comparisons for a teacher-student setup with $d = 100$, $p = 50$, and $n = 150$. Several similar experiments for different configurations are available in our provided notebook. It is observed that with $\kappa = 1$, the generalization error does not follow a double descent curve. Recall that $\kappa = 1$ implies that all the features are learned at the same rate. However, by increasing the value of κ, double descent curves are observed. Very large values of κ imply that some features are practically non-learnable and hence a typical overfitting curve is observed.

4.3.2. The Phase diagram

To further investigate the transition between the two phases of *classical single descent* and *double descent*, we explore the phase diagram. Recall that with Eq. 4.2.6, one can fully characterize the evolution of the generalization dynamics in terms of two scalar variables instead of the d-dimensional parameter space. R and Q presented in Eq. 4.2.7 are macroscopic variables where R represents **the alignment between the teacher and the student** and Q is the **student's (modulated) norm**. Hence, a better generalization performance is achieved with larger R and smaller Q.

The quantities R and Q are not free parameters and both depend on the training dynamics through Eqs. 4.2.14 and 4.2.15. Nevertheless, it is instructive to visualize the generalization error for all pairs of (R, Q). In Figure 4, we visualize the RQ-plane for $(R, Q) \in [0.0, 1.0] \times Q \in [0.0, 1.2]$. At the time of initialization, $(R, Q) = (0, 0)$ as the models are initialized at the origin. As training

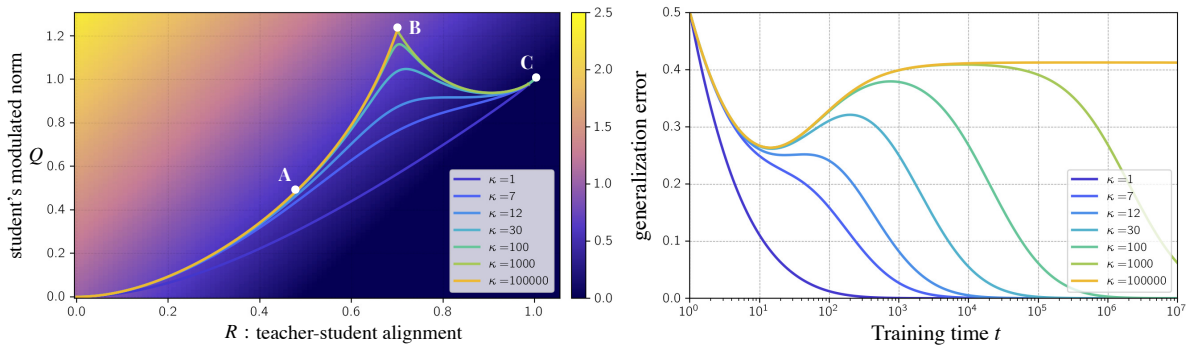


Fig. 4. Left: Phase diagram of the generalization error as a function of $R(t)$ and $Q(t)$ (Eqs. 4.2.14 and 4.2.15). The generalization error for all pairs of $(R, Q) \in [0.0, 1.0] \times [0.0, 1.2]$ is contour-plotted in the background, with the best generalization performance being attained on the lower right part of the plot. The trajectories describe the evolution of $R(t)$ and $Q(t)$ as training proceeds. Each trajectory correspond to a different κ, the condition number of the modulation matrix F in Eq. 4.2.2. κ describes the ratio of the rates at which two sets of features are learned. **Right**: The corresponding generalization curves. **Analysis**: The trajectory with $\kappa = 1e5$ starts at the origin and advances towards point A (a descent in generalization error). Then by over-training, it converges to point B (an ascent). For the other trajectories with smaller κ, a first descent occurs up to the point A, then an ascent happens, but they no longer converge to point B. Instead, by further training, these trajectories converge to point C implying a second descent.

time proceeds, values of R and Q follow the depicted trajectories. In Figure 4, different trajectories correspond to different values of κ, the condition number of the modulation matrix F in Eq. 4.2.2. It is important to note that *the closer a trajectory is to the lower-right, the better the generalization error gets.*

The yellow curve corresponds to the case with large $\kappa = 1e5$, meaning that a subset of features are extremely slower than the others that practically do not get learned. In that case, generalization error exhibits traditional overfitting due to over-training. On the phase diagram, the yellow trajectory starts at $(0, 0)$ and moves towards Point A which has the lowest generalization error of this curve. Then as the training continues, Q increases and as $t \to \infty$ the trajectory lands at Point B which has the worse generalization error (highly-overfitted). Other curves follow the case of $\kappa = 1e5$ up to the vicinity of Point B, but then the trajectories slowly incline towards another fixed point, Point C signalling a second descent in the generalization error.

The phase diagram along with the corresponding generalization curves in Figure 4 illustrate that features that are learned on a faster time-scale are responsible for the initial conventional U-shaped generalization curve, while the second descent can be attributed to the features that are learned at a slower time-scale.

4.3.3. Qualitative comparison with ResNet on Cifar-10

We train a ResNet-18 (He et al., 2016) with layer widths $[64, 2 \times 64, 4 \times 64, 8 \times 64]$. We follow the training setup of Nakkiran et al. (2019a); label noise with a probability 0.15 randomly assign an

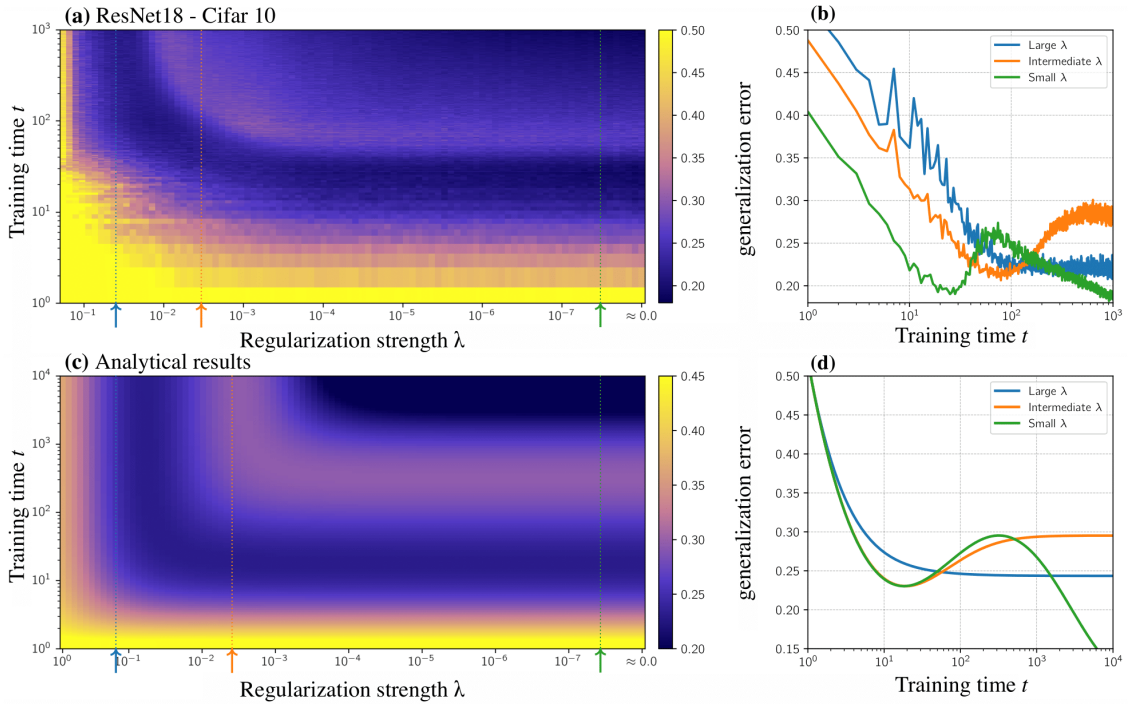


Fig. 5. A qualitative comparison between a ResNet-18 and our analytical results. (a): Heat-map of empirical generalization error (0-1 classification error) for the ResNet-18 trained on Cifar-10 with 15% label noise. X-axis denotes the inverse of weight-decay regularization strength and Y-axis represents the training time. **(c)**: Heat-map of the analytical generalization error (mean squared error) for the linear teacher-student setup with $\kappa = 100$, the condition number of the modulation matrix. **(b, d)**: Three slices of the heat-maps for large, intermediate, and small amounts of regularization. **Analysis**: As predicted by Eqs. 4.2.14 and 4.2.15, $\kappa = 100$ implies that a subset of features are learned 100 times faster that the rest. Intuitively, large amounts of regularization (↑) allow for the fast-learning features to be learned but cause overfitting. Intermediate levels of regularization (↑) result in a classical U-shaped generalization curve but prevent learning of slow features. Small amounts of regularization (↑) allow for both fast and slow features to be learned, leading to a double descent curve.

incorrect label to training examples. Noise is sampled only once before the training starts. We train using Adam (Kingma and Ba, 2014) with learning rate of $1e-4$ for 1K epochs. Experiments are averaged over 50 random seeds.

We conduct an experiment on the classification task of Cifar-10 (Krizhevsky et al., 2009) with varying amount of weight decay regularization strength λ. We monitor the generalization error (0-1 test error) during the course of training and visualize a heat-map of the generalization error for different λ's in Figure 5 (a).

We also conduct a similar experiment with the teacher-student setup presented in Section 4.2.1. We visualize a heat-map of the generalization error which is the mean squared error (MSE) over test distribution in Figure 5 (c). Particularly, we plot Eqs. 4.2.14 and 4.2.15 with a $\kappa = 100$. It is observed that in both experiments, a model with intermediate levels of regularization displays a

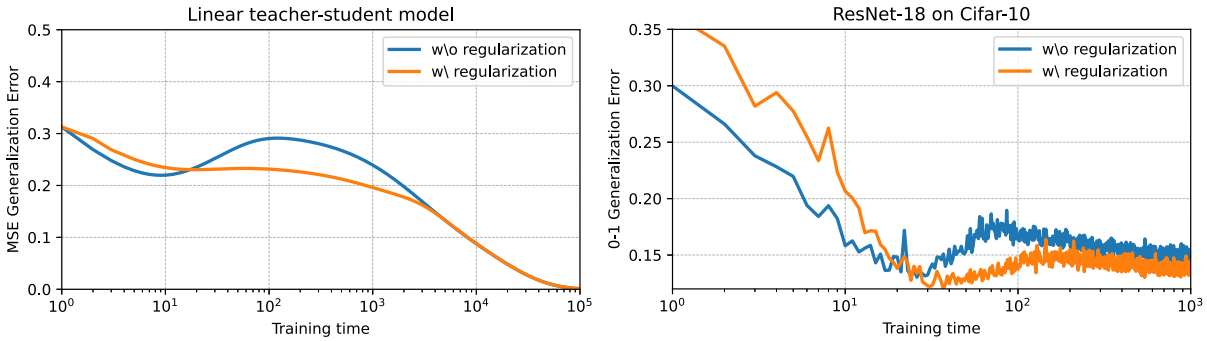


Fig. 6. The effect of regularizing the quantity Q on the generalization curve. Two setups with (w/) and without (w/o) regularization are compared. Both the linear teacher-student model and a ResNet-18 on a binary Cifar-10 benefit from such regularization as the temporary overfitting is diminished. In accordance with Lemma 4.3.1, Q regularization is implemented by simply penalizing the norm of the model's output.

typical overfitting behavior where the generalization error decreases first and then overfits. This is consistent with Eq. .2.36 of the appendix: The amount of regularization λ, is inversely proportional to the training time t implying that larger amounts of regularization act as early stopping.

4.3.4. Diminishing the temporary overfitting

The phase diagram in Figure 4 along with Eq. 4.2.6 suggest that an inflation in the value Q is responsible for the temporary overfitting observed in epoch-wise double descent. As an illustrative experiment, if we could diminish this temporary overfitting, we could expect to observe a *single descent* rather than a double descent curve. To that end, a natural solution is to penalize Q during training. To do that, we introduce the following lemma.

Lemma 4.3.1. *For a linear/linearized model, penalizing Q amounts to adding the following regularizer to the loss,*

$$\mathcal{L}_T \leftarrow \mathcal{L}_T + \alpha ||\hat{y}||^2, \tag{4.3.1}$$

previously introduced in Pezeshki et al. (2020). (See App. .2.5 for Proof).

Figure 6 depicts the effect of this regularizer on the generalization curve. Both linear teacher-student model and ResNet-18 show curves in which the overfitting cusps are diminished. The ResNet experiment is on a binary classification version of the Cifar dataset.

We note that, for any linear model $\hat{y} = Xw$, the regularization $||\hat{y}||^2$ translates to an L2 regularization on the weights that is scaled by the input covariance matrix, as $||\hat{y}||^2 = w^T X^T X w$. Therefore, such regularization slows down the learning along the direction of faster features and hence attempts to equalize the learning scale of different features. We should highlight that mitigating double descent is not the purpose of our work and this experiment is presented to support that the findings from a linear model can still carry over to non-linear networks.

4.4. Related Work and Discussion

Although the term *double descent* has been introduced rather recently (Belkin et al., 2019b), similar behaviors had already been observed and studied in several decades-old works form a statistical physics perspective (Krogh and Hertz, 1992a; Opper, 1995; Opper and Kinzel, 1996; Bös, 1998). More recently, these behaviors have been investigated in the context of modern machine learning, both from an empirical (Amari et al., 2020; Yang et al., 2020) and theoretical perspectives (Geiger et al., 2019; d'Ascoli et al., 2021; Geiger et al., 2020).

Hastie et al. (2019); Advani and Saxe (2017b); Belkin et al. (2020) use random matrix theory (RMT) tools to characterize the asymptotic generalization behavior of over-parameterized linear and random feature models. Mei and Montanari (2019a) extend the same analysis to a random feature model and theoretically derive the model-wise double descent curve for a model with Tikhonov regularization. Jacot et al. (2020) also study double descent in ridge estimators and show an equivalence to kernel ridge regression.

While most of the related work study the non-monotonicity of the generalization error as a function of the model size or sample size, Nakkiran et al. (2019a) introduced the epoch-wise double descent, where the double descent occurs as the training time increases. There has been limited work on studying of epoch-wise double descent. Very recently, Heckel and Yilmaz (2020) and Stephenson and Lee (2021) have focused on finding the roots of this phenomenon.

Heckel and Yilmaz (2020) provides *upper bounds* on the risk of single and two layer models in a regression setting where the input data has distinct feature variances. Heckel and Yilmaz (2020) demonstrate that a superposition of two or more bias-variance tradeoff curves leads to epoch-wise double descent. The authors also show that different layers of the network are learned at different epochs. For that reason, epoch-wise double descent can be eliminated by appropriate selection of learning rates for individual weights. Stephenson and Lee (2021) arrive at similar conclusions. A data model is constructed so that the noise is explicitly added *only* to the fast-learning features while slow-learning features remain noise-free. Consequently, the noisy features form a U-shaped generalization curve while noiseless but slow features are responsible for the second descent.

Our findings and those of Heckel and Yilmaz (2020) and Stephenson and Lee (2021) reinforce one another with a common central finding that the epoch-wise double descent results from different features/layers being learned at different time-scales. However, we also highlight that both Heckel and Yilmaz (2020) and Stephenson and Lee (2021) use tools from random matrix theory to study distinct data models from our teacher-student setup. We study a similar phenomenon by leveraging the replica method from statistical physics to characterize the generalization behavior using a set of informative macroscopic parameters. The key novel contribution from our approach is the derivation of the macroscopic quantities R and Q (see Eq. 4.2.7) which track teacher-student alignment, and the student's modulated norm, respectively. Crucially, these quantities can be used to study other generalization phenomena and/or to modify the learning dynamics via their explicit regularization

as illustrated in Section 4.3.4.

We believe our framework sets the stage for further understanding of generalization dynamics beyond the double descent. A future direction to study is a case in which the first descent is strong enough to bring down the training loss to zero such that learning slower features is practically impossible (Pezeshki et al., 2020) or happens after a very large number of epochs (Power et al., 2021). *Grokking* is an instance of such behavior reported by Power et al. (2021) in which the model abruptly learns to perfectly generalize but long after the training loss has reached very small values.

Finally, we note that while our simple teacher-student setup exhibits the epoch-wise double descent, its simplicity introduces several limitations. Studying finer details of the dynamics of neural networks requires more precise, non-linear, and multi-layered models, which introduce novel challenges that remain to be studied in future work.

Chapter 5

Prologue to Second Article

5.1. Article Details

Gradient starvation: A learning proclivity in neural networks. Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, Guillaume Lajoie. Proceeding of Advances in Neural Information Processing Systems (NeurIPS) 2021.

Personal Contributions. The original idea of gradient starvation (GS) was brought up in meetings between Mohammad Pezeshki, Aaron Courville, and Yoshua Bengio. Guillaume Lajoie later proposed a dynamical systems approach in formalizing the phenomenon and Doina Precup contributed to several discussions on solving it. Mohammad Pezeshki proposed the specific dual-form analysis, contributed to the general idea of theorem 1 (existence of GS) and proved theorem 2 (the solution to GS). Mohammad Pezeshki conducted the experiments. Sekou Oumar Kaba contributed to designing and proving theorem 1 and assisted in writing. Guillaume Lajoie and Mohammad Pezeshki equally contributed in writing. All the authors later revised the paper and provided feedback in several rounds.

5.2. Context

This project was inspired by a series of observations showing that neural networks tend to latch more onto superficial features than the true transferable features. We noticed that in either case, the training loss is zero implying the lack of incentive for further learning. To understand and formalize these observations, we looked into the learning dynamics of linearized models under gradient descent.

Although there is a rich literature on the learning dynamics of neural networks, most of them focus on the particular mean square error (MSE) loss which has linear derivatives and hence simpler learning dynamics. However, the common loss function for classification in neural networks is cross-entropy. Cross-entropy despite being convex has infinite solutions when the data is linearly separable. This work summarizes our efforts at characterizing different zero-loss solutions, with

cross-entropy loss as well as developing a method to favor more generalizable solutions.

5.3. Contributions

This work identifies and formalizes *Gradient Starvation*, a phenomenon that arises when the cross-entropy loss is minimized by capturing only a subset of features relevant for the task, despite the presence of other predictive features that fail to be discovered. This work provides a theoretical explanation for the emergence of such feature imbalances in neural networks. Based on the proposed formalism, we develop guarantees for a novel but simple regularization method aimed at decoupling feature learning dynamics, improving accuracy and robustness in cases hindered by gradient starvation. We illustrate our findings with simple and real-world out-of-distribution (OOD) generalization experiments.

Impact. This work has received a considerable amount of attention and has been cited in several follow-up papers on OOD generalization. Independent studies, such as Galstyan et al. (2021); Pohjonen et al. (2021); Shrestha et al. (2022), have compared our proposed method called spectral decoupling (SD) against several other regularization methods. Pohjonen et al. (2021) validates the effectiveness of SD for bias mitigation and states *"We recommend using spectral decoupling as an implicit bias mitigation method in any neural network intended for clinical use"* as clinical datasets often suffer from significant implicit biases.

Chapter 6

Gradient Starvation: A Learning Proclivity in Neural Networks

6.1. Introduction

In 1904, a horse named *Hans* attracted worldwide attention due to the belief that it was capable of doing arithmetic calculations (Pfungst, 1911). Its trainer would ask Hans a question, and Hans would reply by tapping on the ground with its hoof. However, it was later revealed that the horse was only noticing subtle but distinctive signals in its trainer's unconscious behavior, unbeknown to him, and not actually performing arithmetic. An analogous phenomenon has been noticed when training neural networks (e.g. Ribeiro et al., 2016; Zhao et al., 2017; Jo and Bengio, 2017; Heinze-Deml and Meinshausen, 2017; Belinkov and Bisk, 2017; Baker et al., 2018; Gururangan et al., 2018; Jacobsen et al., 2018; Zech et al., 2018; Niven and Kao, 2019; Ilyas et al., 2019; Brendel and Bethge, 2019; Lapuschkin et al., 2019; Oakden-Rayner et al., 2020). In many cases, state-of-the-art neural networks appear to focus on low-level **superficial correlations**, rather than more abstract and robustly informative features of interest (Beery et al., 2018; Rosenfeld et al., 2018; Hendrycks and Dietterich, 2019; McCoy et al., 2019; Geirhos et al., 2020).

The rationale behind this phenomenon is well known by practitioners: given strongly-correlated and fast-to-learn features in training data, gradient descent is biased towards learning them first. However, the precise conditions leading to such learning dynamics, and how one might intervene to control this *feature imbalance* are not entirely understood. Recent work aims at identifying the reasons behind this phenomenon (Valle-Pérez et al., 2018; Nakkiran et al., 2019b; Cao et al., 2019; Nar and Sastry, 2019; Jacobsen et al., 2018; Niven and Kao, 2019; Wang et al., 2019; Shah et al., 2020; Rahaman et al., 2019; Xu et al., 2019b; Hermann and Lampinen, 2020; Parascandolo et al., 2020; Ahuja et al., 2020b), while complementary work quantifies resulting shortcomings, including poor generalization to out-of-distribution (OOD) test data, reliance upon spurious correlations, and lack of robustness (Geirhos et al., 2020; McCoy et al., 2019; Oakden-Rayner et al., 2020; Hendrycks and Gimpel, 2016; Lee et al., 2018; Liang et al., 2017; Arjovsky et al., 2019). However most

established work focuses on squared-error loss and its particularities, where results do not readily generalize to other objective forms. This is especially problematic since for several classification applications, cross-entropy is the loss function of choice, yielding very distinct learning dynamics. In this paper, we argue that *Gradient Starvation*, first coined in Combes et al. (2018), is a leading cause for this *feature imbalance* in neural networks trained with cross-entropy, and propose a simple approach to mitigate it.

Here we summarize our contributions:

- We provide a theoretical framework to study the learning dynamics of linearized neural networks trained with cross-entropy loss in a dual space.
- Using perturbation analysis, we formalize Gradient Starvation (GS) in view of the coupling between the dynamics of orthogonal directions in the feature space (Thm. 6.3.5).
- We leverage our theory to introduce Spectral Decoupling (SD) (Eq. 6.3.17) and prove this simple regularizer helps to decouple learning dynamics, mitigating GS.
- We support our findings with extensive empirical results on a variety of classification and adversarial attack tasks. All code and experiment details available at `GitHub repository`.

In the rest of the paper, we first present a simple example to outline the consequences of GS. We then present our theoretical results before outlining a number of numerical experiments. We close with a review of related work followed by a discussion.

6.2. Gradient Starvation: A simple example

Consider a 2-D classification task with a training set consisting of two classes, as shown in Figure 1. A two-layer ReLU network with 500 hidden units is trained with cross-entropy loss for two different arrangements of the training points. The difference between the two arrangements is that, in one setting, the data is not linearly separable, but a slight shift makes it linearly separable in the other setting. This small shift allows the network to achieve a negligible loss by only learning to discriminate along the horizontal axis, ignoring the other. This contrasts with the other case, where both features contribute to the learned classification boundary, which arguably matches the data structure better. We observe that training longer or using different regularizers, including weight decay (Krogh and Hertz, 1992b), dropout (Srivastava et al., 2014), batch normalization (Ioffe and Szegedy, 2015), as well as changing the optimization algorithm to Adam (Kingma and Ba, 2014) or changing the network architecture or the coordinate system, do not encourage the network to learn a curved decision boundary. (See App. .4 for more details.)

We argue that this occurs because cross-entropy loss leads to gradients "starved" of information from vertical features. Simply put, when one feature is learned faster than the others, the gradient contribution of examples containing that feature is diminished (i.e., they are correctly processed based on that feature alone). This results in a lack of sufficient gradient signal, and hence prevents any remaining features from being learned. This simple mechanism has potential consequences,

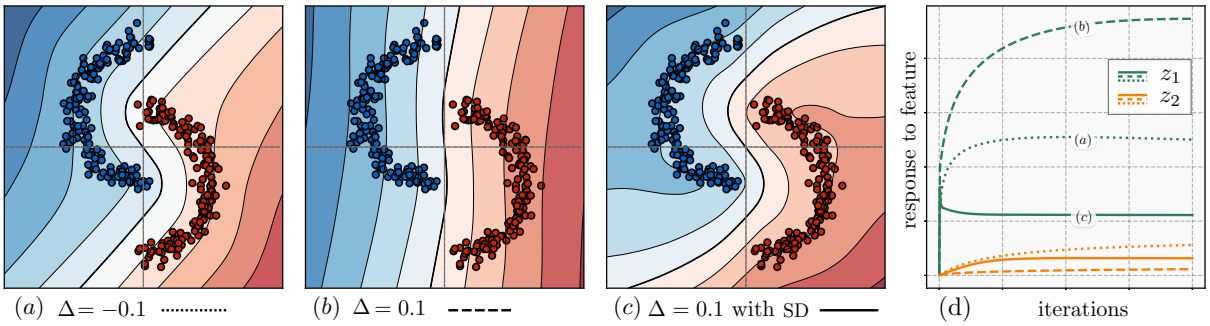


Fig. 1. Diagram illustrating the effect of gradient starvation in a simple 2-D classification task. **(a)** Data is not linearly separable and the learned decision boundary is curved. **(b)** Data is linearly separable by a small margin ($\Delta = 0.1$). This small margin allows the network to discriminate confidently only along the horizontal axis and ignore the vertical axis. **(c)** Data is linearly separable as in (b). However, with the proposed Spectral decoupling (SD), a curved decision boundary with a large margin is learned. **(d)** Diagram shows the evolution of two of the features (Eq. 6.3.4) of the dynamics in three cases shown as dotted, dashed and solid lines. **Analysis:** (dotted) vs (dashed): Linear separability of the data results in an increase in z_1 and a decrease (starvation) of z_2. (dashed) vs (solid): SD suppresses z_1 and hence allows z_2 to grow. Decision boundaries are averaged over ten runs. More experiments with common regularization methods are provided in App. .4.

which we outline below.

Lack of robustness. In the example above, even in the right plot, the training loss is nearly zero, and the network is very confident in its predictions. However, the decision boundary is located very close to the data points. This could lead to adversarial vulnerability as well as lack of robustness when generalizing to out-of-distribution data.

Excessive invariance. GS could also result in neural networks that are invariant to task-relevant changes in the input. In the example above, it is possible to obtain a data point with low probability under the data distribution, but that would still be classified with high confidence.

Implicit regularization. One might argue that according to Occam's razor, a simpler decision boundary should generalize better. In fact, if both training and test sets share the same dominant feature (in this example, the feature along the horizontal axis), GS naturally prevents the learning of less dominant features that could otherwise result in overfitting. Therefore, depending on our assumptions on the training and test distributions, GS could also act as an implicit regularizer. We provide further discussion on the *implicit regularization* aspect of GS in Section 6.5.

6.3. Theoretical Results

In this section, we study the learning dynamics of neural networks trained with cross-entropy loss. Particularly, we seek to decompose the learning dynamics along orthogonal directions in the feature space of neural networks, to provide a formal definition of GS, and to derive a simple regularization method to mitigate it. For analytical tractability, we make three key assumptions: (1) we study deep networks in the Neural Tangent Kernel (NTK) regime, (2) we treat a binary classification task, (3) we decompose the interaction between two features. In Section 8.5, we

demonstrate our results hold beyond these simplifying assumptions, for a wide range of practical settings. All derivation details can be found in SM .5.

6.3.1. Problem Setup and Gradient Starvation Definition

Let $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ denote a training set containing n datapoints with d dimensions, where, $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n] \in \mathbb{R}^{n \times d}$ and their corresponding class label $\mathbf{y} \in \{-1, +1\}^n$. Also let $\hat{\mathbf{y}}(\mathbf{X}) := f^{(L)}(\mathbf{X}) : \mathbb{R}^{n \times d} \to \mathbb{R}^n$ represent the logits of an L-layer fully-connected neural network where each hidden layer $h^{(l)}(x) \in \mathbb{R}^{d_l}$ is defined as follows,

$$\begin{cases} f^{(l)}(\mathbf{x}_i) = \mathbf{W}^{(l)} h^{(l-1)}(\mathbf{x}_i) \\ h^{(l)}(\mathbf{x}_i) = \sqrt{\frac{\gamma}{d_l}} \xi(f^{(l)}(\mathbf{x}_i)) \end{cases}, \; l \in \{0, 1, ..., L\}, \tag{6.3.1}$$

in which $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ is a weight matrix drawn from $\mathcal{N}(0, \mathbf{I})$ and γ is a scaling factor to ensure that norm of each $h^{(l-1)}$ is preserved at initialization (See Du et al. (2018a) for a formal treatment). The function $\xi(.)$ is also an element-wise non-linear activation function.

Let $\boldsymbol{\theta} = \text{concat}\Big(\cup_{l=1}^{L} \text{vec}(\mathbf{W}^{(l)}) \Big) \in \mathbb{R}^m$ be the concatenation of all vectorized weight matrices with m as the total number of parameters. In the NTK regime Jacot et al. (2018), in the limit of infinite width, the output of the neural network can be approximated as a linear function of its parameters governed by the neural tangent random feature (NTRF) matrix Cao and Gu (2019),

$$\boldsymbol{\Phi}(\mathbf{X}, \boldsymbol{\theta}) = \frac{\partial \hat{\mathbf{y}}(\mathbf{X}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{n \times m}. \tag{6.3.2}$$

In the wide-width regime, the NTRF changes very little during training Lee et al. (2019), and the output of the neural network can be approximated by a first order Taylor expansion around the initialization parameters $\boldsymbol{\theta}_0$. Setting $\boldsymbol{\Phi}_0 \equiv \boldsymbol{\Phi}(\mathbf{X}, \boldsymbol{\theta}_0)$ and then, without loss of generality, centering parameters and the output coordinates to their value at the initialization ($\boldsymbol{\theta}_\mathbf{0}$ and $\hat{\mathbf{y}}_0$), we get

$$\hat{\mathbf{y}}(\mathbf{X}, \boldsymbol{\theta}) = \boldsymbol{\Phi}_0 \boldsymbol{\theta}. \tag{6.3.3}$$

Dominant directions in the feature space as well as the parameter space are given by principal components of the NTRF matrix $\boldsymbol{\Phi}_0$, which are the same as those of the NTK Gram matrix (Yang and Salman, 2019). We therefore introduce the following definition.

Definition 6.3.1 (Features and Responses)**.** Consider the singular value decomposition (SVD) of the matrix $\mathbf{Y}\boldsymbol{\Phi}_0 = \mathbf{U}\mathbf{S}\mathbf{V}^T$, where $\mathbf{Y} = \text{diag}(\mathbf{y})$. The jth **feature** is given by $(\mathbf{V}^T)_{j.}$. The **strength** of jth feature is represented by $s_j = (\mathbf{S})_{jj}$. Also, $(\mathbf{U})_{.j}$ contains the weights of this feature in all examples. A neural network's **response to a feature** j is given by z_j where,

$$\mathbf{z} := \mathbf{U}^T \mathbf{Y} \hat{\mathbf{y}} = \mathbf{S}\mathbf{V}^T \boldsymbol{\theta}. \tag{6.3.4}$$

In Eq. 6.3.4, the response to feature j is the sum of the responses to every example in $(\mathbf{Y}\hat{\mathbf{y}})$ multiplied by the weight of the feature in that example ($\mathbf{U}^T$). For example, if all elements of $(\mathbf{U})_{.j}$

are positive, it indicates a perfect correlation between this feature and class labels. We are now equipped to formally define GS.

Definition 6.3.2 (Gradient Starvation)**.** Recall the the model prescribed by Eq. 6.3.3. Let z_j^* denote the model's response to feature j at training optimum $\boldsymbol{\theta}^*$[1]. Feature i **starves the gradient** for feature j if $dz_j^*/d(s_i^2) < 0$.

This definition of GS implies that an increase in the strength of feature i has a detrimental effect on the learning of feature j. We now derive conditions for which learning dynamics of system 6.3.3 suffer from GS.

6.3.2. Training Dynamics

We consider the widely used ridge-regularized cross-entropy loss function,

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbf{1} \cdot \log\left[1 + \exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2, \tag{6.3.5}$$

where $\mathbf{1}$ is a vector of size n with all its elements equal to 1. This vector form simply represents a summation over all the elements of the vector it is multiplied to. $\lambda \in [0, \infty)$ denotes the weight decay coefficient.

Direct minimization of this loss function using the gradient descent obeys coupled dynamics and is difficult to treat directly (Combes et al., 2018). To overcome this problem, we call on a variational approach that leverages the Legendre transformation of the loss function. This allows tractable dynamics that can directly incorporate rates of learning in different feature directions. Following (Jaakkola and Haussler, 1999), we note the following inequality,

$$\log\left[1 + \exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] \geq H(\boldsymbol{\alpha}) - \boldsymbol{\alpha} \odot \mathbf{Y}\hat{\mathbf{y}}, \tag{6.3.6}$$

where $H(\boldsymbol{\alpha}) = -\left[\boldsymbol{\alpha}\log\boldsymbol{\alpha} + (1-\boldsymbol{\alpha})\log(1-\boldsymbol{\alpha})\right]$ is Shannon's binary entropy function, $\boldsymbol{\alpha} \in (0,1)^n$ is a variational parameter defined for each training example, and $\odot$ denotes the element-wise vector product. Crucially, the equality holds when the maximum of r.h.s. w.r.t $\boldsymbol{\alpha}$ is achieved at $\boldsymbol{\alpha}^* = \frac{\partial \mathcal{L}}{\partial(\mathbf{Y}\hat{\mathbf{y}})^T}$, which leads to the following optimization problem,

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\theta) = \min_{\boldsymbol{\theta}} \max_{\boldsymbol{\alpha}} \left(\mathbf{1} \cdot H(\boldsymbol{\alpha}) - \boldsymbol{\alpha}\mathbf{Y}\hat{\mathbf{y}} + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2\right), \tag{6.3.7}$$

where the order of min and max can be swapped (see Lemma 3 of Jaakkola and Haussler (1999)). Since the neural network's output is approximated by a linear function of $\boldsymbol{\theta}$, the minimization can be performed analytically with an critical value $\boldsymbol{\theta}^{*T} = \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{Y}\boldsymbol{\Phi}_0$, given by a weighted sum of the training examples. This results in the following maximization problem on the dual variable, i.e., $\min_\theta \mathcal{L}(\theta)$ is equivalent to,

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \max_{\boldsymbol{\alpha}} \left(\mathbf{1} \cdot H(\boldsymbol{\alpha}) - \frac{1}{2\lambda}\boldsymbol{\alpha}\mathbf{Y}\boldsymbol{\Phi}_0\boldsymbol{\Phi}_0^T\mathbf{Y}^T\boldsymbol{\alpha}^T\right). \tag{6.3.8}$$

[1]Training optimum refers to the solution to $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) = 0$.

By applying continuous-time gradient ascent on this optimization problem, we derive an autonomous differential equation for the evolution of $\boldsymbol{\alpha}$, which can be written in terms of features (see Definition 6.3.1),

$$\dot{\boldsymbol{\alpha}} = \eta \left(-\log \boldsymbol{\alpha} + \log \left(\mathbf{1} - \boldsymbol{\alpha} \right) - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \mathbf{S}^2 \mathbf{U}^T \right), \tag{6.3.9}$$

where η is the learning rate (see App. .5.1 for more details). For this dynamical system, we see that the logarithm term acts as barriers that keep $\alpha_i \in (0,1)$. The other term depends on the matrix $\mathbf{U}\mathbf{S}^2\mathbf{U}^T$, which is positive definite, and thus pushes the system towards the origin and therefore drives learning.

When $\lambda \ll s_k^2$, where k is an index over the singular values, the linear term dominates Eq. 6.3.9, and the fixed point is drawn closer towards the origin. Approximating dynamics with a first order Taylor expansion around the origin of the second term in Eq. 6.3.9, we get

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \left(\mathbf{S}^2 + \lambda \mathbf{I} \right) \mathbf{U}^T \right), \tag{6.3.10}$$

with stability given by the following theorem with proof in App. .5.

Theorem 6.3.3. *Any fixed points of the system in Eq. 6.3.10 is attractive in the domain* $\alpha_i \in (0,1)$.

At the fixed point $\boldsymbol{\alpha}^*$, corresponding to the optimum of Eq. 6.3.8, the feature response of the neural network is given by,

$$\mathbf{z}^* = \frac{1}{\lambda} \mathbf{S}^2 \mathbf{U}^T \boldsymbol{\alpha}^{*T}. \tag{6.3.11}$$

See App. .3 for further discussions on the distinction between "feature space" and "parameter space". Below, we study how the strength of one feature could impact the response of the network to another feature which leads to GS.

6.3.3. Gradient Starvation Regime

In general, we do not expect to find an analytical solution for the dynamics of the coupled non-linear dynamical system of Eq. 6.3.10. However, there are at least two cases where a decoupled form for the dynamics allows to find an exact solution. We first introduce these cases and then study their perturbation to outline general lessons.

(1) If the matrix of singular values $\mathbf{S}^2$ is proportional to the identity: This is the case where all the features have the same strength s^2. The fixed points are then given by,

$$\alpha_i^* = \frac{\lambda \mathcal{W}(\lambda^{-1} s^2 + 1)}{s^2 + \lambda}, \qquad z_j^* = \frac{s^2 \mathcal{W}(\lambda^{-1} s^2 + 1)}{s^2 + \lambda} \sum_i u_{ij}, \tag{6.3.12}$$

where $\mathcal{W}$ is the Lambert W function.

(2) If the matrix $\mathbf{U}$ is a permutation matrix: This is the case in which each feature is associated with a single example only. The fixed points are then given by,

$$\alpha_i^* = \frac{\lambda \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}, \qquad z_j^* = \frac{s_i^2 \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}. \tag{6.3.13}$$

To study a minimal case of starvation, we consider a variation of case 2 with the following assumption which implies that each feature is not associated with a single example anymore.

Lemma 6.3.4. *Assume* $\mathbf{U}$ *is a perturbed identity matrix (a special case of a permutation matrix) in which the off-diagonal elements are proportional to a small parameter* $\delta > 0$*. Then, the fixed point of the dynamical system in Eq. 6.3.10 can be approximated by,*

$$\boldsymbol{\alpha}^* = (1 - \log(\boldsymbol{\alpha}_0^*)) \left[\mathbf{A} + diag\left(\boldsymbol{\alpha}_0^{*-1} \right) \right]^{-1}, \tag{6.3.14}$$

where $\mathbf{A} = \lambda^{-1} \mathbf{U}(\mathbf{S^2} + \lambda \mathbf{I}) \mathbf{U}^T$ *and* $\boldsymbol{\alpha}_0^*$ *is the fixed point of the uncoupled system with* $\delta = 0$*.*

For sake of ease of derivations, we consider the two dimensional case where,

$$\mathbf{U} = \begin{pmatrix} \sqrt{1 - \delta^2} & -\delta \\ \delta & \sqrt{1 - \delta^2} \end{pmatrix}, \tag{6.3.15}$$

which is equivalent to a U matrix with two blocks of features with no intra-block coupling and δ amount of inter-block coupling.

Theorem 6.3.5 (Gradient Starvation Regime)**.** *Consider a neural network in the linear regime, trained under cross-entropy loss for a binary classification task. With definition 6.3.1, assuming coupling between features 1 and 2 as in Eq. 6.3.15 and* $s_1^2 > s_2^2$*, we have,*

$$\frac{\mathrm{d}z_2^*}{\mathrm{d}s_1^2} < 0, \tag{6.3.16}$$

which implies GS.

While Thm. 6.3.5 outlines conditions for GS in two dimensional feature space, we note that the same rationale naturally extends to higher dimensions, where GS is defined pairwise over feature directions. For a classification task, Thm. 6.3.5 indicates that gradient starvation occurs when the data admits different feature strengths, and coupled learning dynamics. GS is thus naturally expected with cross-entropy loss. Its detrimental effects however (as outlined in Sect. 6.2) arise in settings with large discrepancies between feature strengths, along with network connectivity that couples these features' directions. This phenomenon readily extends to multi-class settings, and we validate this case with experiments in Sect. 8.5. Next, we introduce a simple regularizer that encourages feature decoupling, thus mitigating GS by insulating strong features from weaker ones.

6.3.4. Spectral Decoupling

By tracing back the equations of the previous section, one may realize that the term $U^T S^2 U$ in Eq. 9 is not diagonal in the general case, and consequently introduces coupling between α_i's and hence, between the features z_i's. We would like to discourage solutions that couple features in this

way. To that end, we introduce a simple regularizer: Spectral Decoupling (SD). SD replaces the general L2 weight decay term in Eq. 6.3.5 with an L2 penalty exclusively on the network's logits, yielding

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbf{1} \cdot \log\left[1 + \exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\|\hat{\mathbf{y}}\|^2. \tag{6.3.17}$$

Repeating the same analysis steps taken above, but with SD instead of general L2 penalty, the critical value for $\boldsymbol{\theta}^*$ becomes $\boldsymbol{\theta}^* = \frac{1}{\lambda}\boldsymbol{\alpha}\boldsymbol{Y}\boldsymbol{\Phi}_0 V\mathbf{S}^{-2}V^T$. This new expression for $\boldsymbol{\theta}^*$ results in the following modification of Eq. 6.3.9,

$$\dot{\boldsymbol{\alpha}} = \eta\left(\log\frac{\mathbf{1}-\boldsymbol{\alpha}}{\boldsymbol{\alpha}} - \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{U}\mathbf{S}^2\mathbf{S}^{-2}\mathbf{U}^T\right) = \eta\left(\log\frac{\mathbf{1}-\boldsymbol{\alpha}}{\boldsymbol{\alpha}} - \frac{1}{\lambda}\boldsymbol{\alpha}\right), \tag{6.3.18}$$

where as earlier, $\log$ and division are taken element-wise on the coordinates of $\boldsymbol{\alpha}$.

Note that in contrast to Eq. 6.3.9 the matrix multiplication involving U and S in Eq. 6.3.18 cancels out, leaving α_i independent of other $\alpha_{j\neq i}$'s. We point out this is true for any initial coupling, without simplifying assumptions. Thus, a simple penalty on output weights promotes decoupled dynamics across the dual parameter α_i's, which track learning dynamics of feature responses (see Eq. 6.3.7). Together with Thm. 6.3.5, Eq. 6.3.18 suggests SD should mitigate GS and promote balanced learning dynamics across features. We now verify this in numerical experiments. For further intuition, we provide a simple experiment, summarized in Fig. 1, where directly visualizes the primal vs. the dual dynamics as well as the effect of the proposed spectral decoupling method.

6.4. Experiments

The experiments presented here are designed to outline the presence of GS and its consequences, as well as the efficacy of our proposed regularization method to alleviate them. Consequently, we highlight that achieving state-of-the-art results is not the objective. For more details including the scheme for hyper-parameter tuning, see App. .4.

6.4.1. Two-Moon classification and the margin

Recall the simple 2-D classification task between red and blue data points in Fig. 1.

Fig. 1 **(c)** demonstrates the learned decision boundary when SD is used. SD leads to learning a curved decision boundary with a larger margin in the input space. See App. .4 for additional details and experiments.

6.4.2. CIFAR classification and adversarial robustness

To study the classification margin in deeper networks, we conduct a classification experiment on CIFAR-10, CIFAR-100, and CIFAR-2 (cats vs dogs of CIFAR-10) (Krizhevsky et al., 2009) using a convolutional network with ReLU non-linearity. Unlike linear models, the margin to a non-linear decision boundary cannot be computed analytically. Therefore, following the approach in Nar et al.

(2019), we use "the norm of input-disturbance required to cross the decision boundary" as a proxy for the margin. The disturbance on the input is computed by projected gradient descent (PGD) (Rauber et al., 2017), a well-known adversarial attack.

Dataset	Method	Train*	Test IID	Test OOD†
Cifar-2	w/o SD	100.0% ± 0.0	95.2% ± 0.12	42.3% ± 3.0
	w/ SD (λ=0.01)	100.0% ± 0.0	95.3% ± 0.17	69.7% ± 2.9
Cifar-10	w/o SD	99.9% ± 0.01	92.8% ± 0.15	30.1% ± 2.1
	w/ SD(λ=0.01)	99.9% ± 0.01	92.9% ± 0.16	67.7% ± 1.5
Cifar-100	w/o SD	99.7% ± 0.01	69.2% ± 0.29	14.3% ± 2.0
	w/ SD(λ=0.05)	99.7% ± 0.02	70.5% ± 0.26	24.9% ± 1.9

† Accuracy (± std) for 10 runs.

Table 1. Table compares adversarial robustness of ERM (vanilla cross-entropy) vs SD with a CNN trained on CIFAR-2, 10, and 100 (setup of Nar et al. (2019)). SD consistently achieves a better OOD performance.

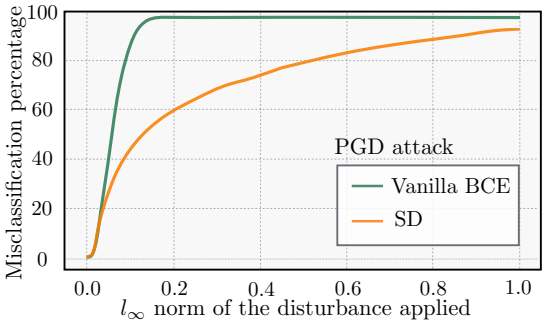


Fig. 2. The plot shows the cumulative distribution function (CDF) of the margin for the CIFAR-2 binary classification. SD appears to improve the margin considerably.

Table 1 includes the results for IID (original test set) and OOD (perturbed test set by $\epsilon_{\text{PGD}} = 0.05$). Fig. 2 shows the percentange of mis-classifications as the norm of disturbance is increased for the Cifar-2 dataset. This plot can be interpreted as the cumulative distribution function (CDF) of the margin and hence a lower curve reads as a more robust network with a larger margin. This experiment suggests that when trained with vanilla cross-entropy, even slight disturbances in the input deteriorates the network's classification accuracy. That is while spectral decoupling (SD) improves the margin considerably. Importantly, this improvement in robustness does not seem to compromise the noise-free test performance. It should also be highlighted that SD does not explicitly aim at maximizing the margin and the observed improvement is in fact a by-product of decoupled learning of latent features. See Section 6.5 for a discussion on why cross-entropy results in a poor margin while being considered a max-margin classifier in the literature (Soudry et al., 2018a).

6.4.3. Colored MNIST with color bias

We conduct experiments on the Colored MNIST Dataset, proposed in Arjovsky et al. (2019). The task is to predict binary labels $y = -1$ for digits 0 to 4 and $y = +1$ for digits 5 to 9. A color channel (red, green) is artificially added to each example to deliberately impose a spurious correlation between the color and the label. The task has three environments:

- Training env. 1: Color is correlated with the labels with 0.9 probability.
- Training env. 2: Color is correlated with the labels with 0.8 probability.
- Testing env.: Color is correlated with the labels with 0.1 probability (0.9 reversely correlated).

Because of the opposite correlation between the color and the label in the test set, only learning to classify based on color would be disastrous at testing. For this reason, Empirical Risk Minimization

(ERM) performs very poorly on the test set (23.7 % accuracy) as shown in Tab. 2.

Method	Train Accuracy	Test Accuracy
ERM (Vanilla Cross-Entropy)	91.1 % (±0.4)	23.7 % (±0.8)
REx (Krueger et al., 2020)	71.5 % (±1.0)	68.7 % (±0.9)
IRM (Arjovsky et al., 2019)	70.5 % (±0.6)	67.1 % (±1.4)
SD (this work)	70.0 % (±0.9)	68.4 % (±1.2)
Oracle - (grayscale images)	73.5 % (±0.2)	73.0 % (±0.4)
Random Guess	50 %	50 %

Table 2. Test accuracy on test examples of the Colored MNIST after training for 1k epochs. The standard deviation over 10 runs is reported in parenthesis. ERM stands for the empirical risk minimization. Oracle is an ERM trained on grayscale images. Note that due to 25 % label noise, a hypothetical optimum achieves 75 % accuracy (the upper bound).

Invariant Risk Minimization (IRM) (Arjovsky et al., 2019) on the other hand, performs well on the test set with (67.1 % accuracy). However, IRM requires access to multiple (two in this case) separate training environments with varying amount of spurious correlations. IRM uses the variance between environments as a signal for learning to be "invariant" to spurious correlations. Risk Extrapolation (REx) (Krueger et al., 2020) is a related training method that encourages learning invariant representations. Similar to IRM, it requires access to multiple training environments in order to quantify the concept of "invariance".

SD achieves an accuracy of 68.4 %. Its performance is remarkable because unlike IRM and REx, SD does not require access to multiple environments and yet performs well when trained on a single environment (in this case the aggregation of both of the training environments).

A natural question that arises is **"How does SD learn to ignore the color feature without having access to multiple environments?"** The short answer is that **it does not**! In fact, we argue that SD learns the color feature but it **also** learns other predictive features, i.e., the digit shape features. At test time, the predictions resulting from the shape features prevail over the color feature. To validate this hypothesis, we study a trained model with each of these methods (ERM, IRM, SD) on four variants of the test environment: 1) grayscale-digits: No color channel is provided and the network should rely on shape features only. 2) colored-digits: Both color and digit are provided however the color is negatively correlated (opposite of the training set) with the label. 3) grayscale-blank: All images are grayscale and blank and hence do not provide any information. 4) colored-blank: Digit features are removed and only the color feature is kept, also with reverse label compared to training. Fig. 3 summarizes the results. For more discussions see App. .4.

As a final remark, we should highlight that, by design, this task assumes access to the test environment for hyperparameter tuning for all the reported methods. This is not a valid assumption in general, and hence the results should be only interpreted as a probe that shows that SD could provide an important level of control over what features are learned.

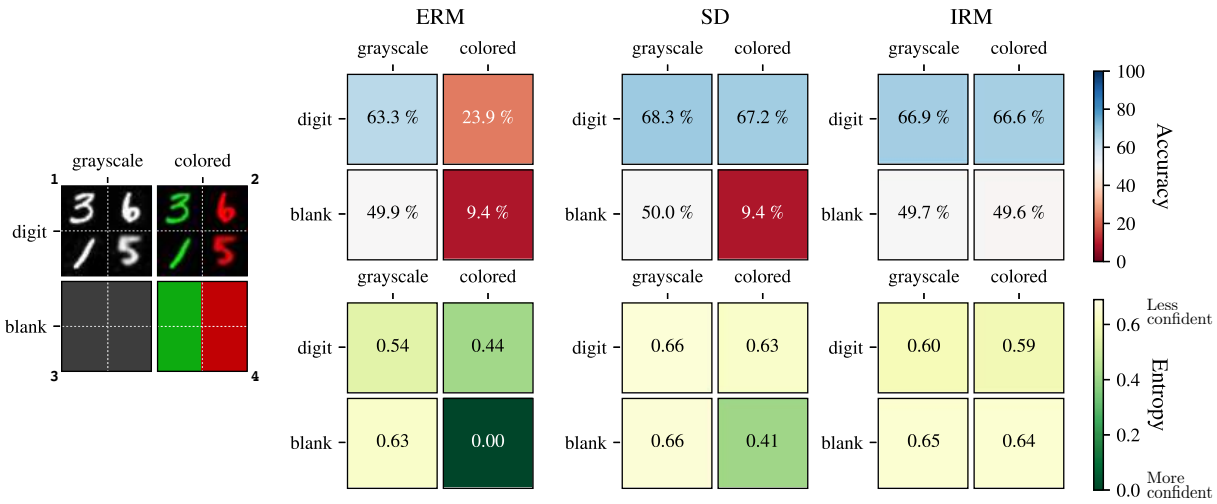


Fig. 3. Diagram comparing ERM, SD, and IRM on four different test environments on which we evaluate a pre-trained model. Top and bottom rows show the accuracy and the entropy (inverse of confidence), respectively. **Analysis:** Compare three values of 9.4 %, 9.4 %, and 49.6 %: Both ERM and SD have learned the color feature but since it is inversely correlated with the label, when only the color feature is provided, as expected both ERM and SD performs poorly. Now compare 0.00 and 0.41: Although both ERM and SD have learned the color feature, ERM is much more confident on its predictions (zero entropy). As a consequence, when digit features are provided along with the color feature (colored-digit environment), ERM still performs poorly (23.9 %) but SD achieves significantly better results (67.2 %). IRM ignores the color feature altogether but it requires access to multiple training environments.

6.4.4. CelebA with gender bias

The CelebA dataset (Liu et al., 2015a) contains 162k celebrity faces with binary attributes associated with each image. Following the setup of (Sagawa et al., 2019), the task is to classify images with respect to their hair color into two classes of blond or dark hair. However, the `Gender` $\in$ {`Male`, `Female`} is spuriously correlated with the `HairColor` $\in$ {`Blond`, `Dark`} in the training data. The rarest group which is blond males represents only 0.85 % of the training data (1387 out of 162k examples). We train a ResNet-50 model (He et al., 2016) on this task. Tab. 3 summarizes the results and compares the performance of several methods. A model with vanilla cross-entropy (ERM) appears to generalize well on average but fails to generalize to the rarest group (blond males) which can be considered as "weakly" out-of-distribution (OOD). Our proposed SD improves the performance more than twofold. It should be highlighted that for this task, we use a variant of SD in which, $\frac{\lambda}{2}||\hat{y}-\gamma||_2^2$ is added to the original cross-entropy loss. The hyper-parameters λ and γ are tuned separately for each class (a total of four hyper-parameters). This variant of SD does provably decouple the dynamics too but appears to perform better than the original SD in Eq. 6.3.17 in this task.

Other proposed methods presented in Tab. 3 also show significant improvements on the performance of the worst group accuracy. The recently proposed "Learning from failure" (LfF) (Nam et al., 2020) achieves comparable results to SD, but it requires simultaneous training of two networks. Group DRO (Sagawa et al., 2019) is another successful method for this task. However,

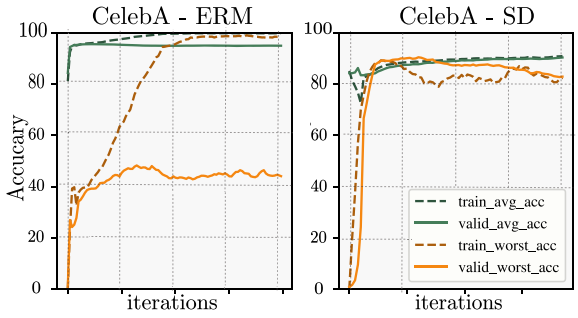


Fig. 4. CelebA: `blond` vs `dark` hair classification. The `HairColor` and the `Gender` are spuriously correlated which leads to poor OOD performance with ERM, however SD significantly improves performance. ERM's worst group accuracy is significantly lower than SD.

Method	Average Acc.	Worst Group Acc.
ERM	94.61 % (±0.67)	40.35 % (±1.68)
SD (this work)	91.64 % (±0.61)	**83.24** % (±2.01)
LfF	N/A	81.24 % (±1.38)
Group DRO*	91.76 % (±0.28)	87.78 % (±0.96)

Table 3. CelebA: `blond` vs `dark` hair classification with spurious correlation. We report test performance over ten runs. SD significantly improves upon ERM. *Group DRO (Sagawa et al., 2019) requires explicit information about the spurious correlation. LfF (Nam et al., 2020) requires simultaneous training of two networks.

unlike SD, Group DRO requires explicit information about the spuriously correlated attributes. In most practical tasks, information about the spurious correlations is not provided and, dependence on the spurious correlation goes unrecognized.[2]

6.5. Related Work and Discussion

Here, we discuss the related work. Due to space constraints, further discussions are in App. .3.

On learning dynamics and Loss Choice

Several works including Saxe et al. (2013b, 2019); Advani and Saxe (2017b); Lampinen and Ganguli (2018) investigate the dynamics of deep linear networks trained with squared-error loss. Different decompositions of the learning process for neural networks have been used: Rahaman et al. (2019); Xu et al. (2019a); Ronen et al. (2019); Xu et al. (2019b) study the learning in the Fourier domain and show that low-frequency functions are learned earlier than high-frequency ones. Saxe et al. (2013b); Advani et al. (2020); Gidel et al. (2019) provide closed-form equations for the dynamics of linear networks in terms of the principal components of the input covariance matrix. More recently, with the introduction of neural tangent kernel (NTK) (Jacot et al., 2018; Lee et al., 2019), a new line of research is to study the convergence properties of gradient descent (e.g. Allen-Zhu et al., 2019b; Mei and Montanari, 2019b; Chizat and Bach, 2018; Du et al., 2018b; Allen-Zhu et al., 2019a; Huang and Yau, 2019; Goldt et al., 2019; Zou et al., 2020; Arora et al., 2019b; Vempala and Wilmes, 2019). Among them, Arora et al. (2019c); Yang and Salman (2019); Bietti and Mairal (2019); Cao et al. (2019) decompose the learning process along the principal components of the NTK. The message in these works is that the training process can be decomposed into independent learning dynamics along the orthogonal directions.

[2]Recall that it took 3 years for the psychologist, Oskar Pfungst, to realize that Clever Hans was not capable of doing any arithmetic.

Most of the studies mentioned above focus on the particular squared-error loss. For a linearized network, the squared-error loss results in linear learning dynamics, which often admit an analytical solution. However, the de-facto loss function for many of the practical applications of neural networks is the cross-entropy. Using the cross-entropy as the loss function leads to significantly more complicated and non-linear dynamics, even for a linear neural network. In this work, our focus was the cross-entropy loss.

On reliance upon spurious correlations and robustness

In the context of robustness in neural networks, state-of-the-art neural networks appear to naturally focus on low-level superficial correlations rather than more abstract and robustly informative features of interest (e.g. Geirhos et al. (2020)). As we argue in this work, Gradient Starvation is likely an important factor contributing to this phenomenon and can result in adversarial vulnerability. There is a rich research literature on adversarial attacks and neural networks' vulnerability (Szegedy et al., 2013; Goodfellow et al., 2014; Ilyas et al., 2019; Madry et al., 2017; Akhtar and Mian, 2018; Ilyas et al., 2018). Interestingly, Nar and Sastry (2019), Nar et al. (2019) and Jacobsen et al. (2018) draw a similar conclusion and argue that "an insufficiency of the cross-entropy loss" causes excessive invariances to predictive features. Perhaps Shah et al. (2020) is the closest to our work in which authors study the simplicity bias (SB) in stochastic gradient descent. They demonstrate that neural networks exhibit extreme bias that could lead to adversarial vulnerability.

On implicit bias

Despite being highly-overparameterized, modern neural networks seem to generalize very well (Zhang et al., 2016). Modern neural networks generalize surprisingly well in numerous machine tasks. This is despite the fact that neural networks typically contain orders of magnitude more parameters than the number of examples in a training set and have sufficient capacity to fit a totally randomized dataset perfectly (Zhang et al., 2016). The widespread explanation is that the gradient descent has a form of implicit bias towards learning simpler functions that generalize better according to Occam's razor. Our exposition of GS reinforces this explanation. In essence, when training and test data points are drawn from the same distribution, the top salient features are predictive in both sets. We conjecture that in such a scenario, by not learning the less salient features, GS naturally protects the network from overfitting.

The same phenomenon is referred to as *implicit bias*, *implicit regularization*, *simplicity bias* and *spectral bias* in several works (Rahaman et al., 2019; Neyshabur et al., 2014; Gunasekar et al., 2017; Neyshabur et al., 2017b; Nakkiran et al., 2019b; Ji and Telgarsky, 2019; Soudry et al., 2018a; Arora et al., 2019a; Arpit et al., 2017; Gunasekar et al., 2018; Poggio et al., 2017; Ma et al., 2018).

As an active line of research, numerous studies have provided different explanations for this phenomenon. For example, Nakkiran et al. (2019b) justifies the implicit bias of neural networks by showing that stochastic gradient descent learns simpler functions first. Baratin et al. (2020); Oymak et al. (2019) suggests that a form of implicit regularization is induced by an alignment between

NTK's principal components and only a few task-relevant directions. Several other works such as Brutzkus et al. (2017); Gunasekar et al. (2018); Soudry et al. (2018a); Chizat and Bach (2018) recognize the convergence of gradient descent to maximum-margin solution as the essential factor for the generalizability of neural networks. It should be stressed that these work refer to the margin in the hidden space and not in the input space as pointed out in Jolicoeur-Martineau and Mitliagkas (2019). Indeed, as observed in our experiments, the maximum-margin classifier in the hidden space can be achieved at the expense of a small margin in the input space.

On Gradient Starvation and no free lunch theorem

The *no free lunch* theorem (Shalev-Shwartz and Ben-David, 2014; Wolpert, 1996) states that "learning is impossible without making assumptions about training and test distributions". Perhaps, the most commonly used assumption of machine learning is the i.i.d. assumption (Vapnik, 1998), which assumes that training and test data are identically distributed. However, in general, this assumption might not hold, and in many practical applications, there are predictive features in the training set that do not generalize to the test set. A natural question that arises is *how to favor generalizable features over spurious features?* The most common approaches include *data augmentation, controlling the inductive biases, using regularizations,* and more recently *training using multiple environments*.

Here, we would like to elaborate on an interesting thought experiment of Parascandolo et al. (2020): Suppose a neural network is provided with a chess book containing examples of chess games with the best movements indicated by a red arrow. The network can take two approaches: 1) learn how to play chess, or 2) learn just the red arrows. Either of these solutions results in zero training loss on the games in the book while only the former is generalizable to new games. With no external knowledge, the network typically learns the simpler solution.

Recent work aims to leverage the invariance principle across several environments to improve robust learning. This is akin to present several chess books to a network, each with markings indicating the best moves for different sets of games. In several studies (Arjovsky et al., 2019; Krueger et al., 2020; Parascandolo et al., 2020; Ahuja et al., 2020a), methods are developed to aggregate information from multiple training environments in a way that favors the generalizable / domain-agnostic / invariant solution. We argue that even with having access to **only one** training environment, there is useful information in the training set that fails to be discovered due to Gradient Starvation. The information on how to actually play chess is already available in any of the chess books. Still, as soon as the network learns the red arrows, the network has no incentive for further learning. Therefore, *learning the red arrows is not an issue per se, but not learning to play chess is.*

Gradient Starvation: friend or foe?

Here, we would like to remind the reader that GS can have both adverse and beneficial consequences. If the learned features are sufficient to generalize to the test data, gradient starvation can be viewed as an implicit regularizer. Otherwise, Gradient Starvation could have an unfavorable effect,

which we observe empirically when some predictive features fail to be learned. A better understanding and control of Gradient Starvation and its impact on generalization offers promising avenues to address this issue with minimal assumptions. Indeed, our Spectral Decoupling method requires an assumption about feature imbalance but not to pinpoint them exactly, relying on modulated learning dynamics to achieve balance.

GS social impact

Modern neural networks are being deployed extensively in numerous machine learning tasks. Our models are used in critical applications such as autonomous driving, medical prediction, and even justice system where human lives are at stake. However, neural networks appear to base their predictions on superficial biases in the dataset. Unfortunately, biases in datasets could be neglected and pose negative impacts on our society. In fact, aour Celeb-A experiment is an example of the existence of such a bias in the data. As shown in the paper, the gender-specific bias could lead to a superficial high performance and is indeed very hard to detect. Our analysis, although mostly on the theory side, could pave the path for researchers to build machine learning systems that are robust to biases and helps towards fairness in our predictions.

6.6. Conclusion

In this paper, we formalized Gradient Starvation (GS) as a phenomenon that emerges when training with cross-entropy loss in neural networks. By analyzing the dynamical system corresponding to the learning process in a dual space, we showed that GS could slow down the learning of certain features, even if they are present in the training set. We derived spectral decoupling (SD) regularization as a possible remedy to GS.

Chapter 7

Prologue to Third Article

7.1. Article Details

Simple data balancing achieves competitive worst-group-accuracy. Badr Youbi Idrissi, Martin Arjovsky, Mohammad Pezeshki, David Lopez-Paz. Proceeding of Conference on Causal Learning and Reasoning (CLeaR) 2022.

Personal contributions. The idea of this work was initially developed when Mohammad Pezeshki was conducting experiments on the CelebA dataset before joining FAIR. This idea later came up in discussions with Badr Youbi Idrissi and David Lopez-Paz during Mohammad's internship at FAIR. Mohammad provided the initial code that validated the idea on the CelebA dataset. Badr led the project and conducted the majority of the experiments. David and Badr designed and implemented the final hands-off experimental framework with automatic dataset downloads and preparation, hyper-parameter search, model selection, and table generation. All the authors contributed to the writing. Mohammad conducted the experiment in Section 8.4 and prepared Figure 1. David Lopez-Paz and Martin Arjovsky supervised the project.

7.2. Context

An established practice in the fields of OoD generalization and causal reasoning is to test and report new methods on datasets that suffer from some form of bias. The most commonly used datasets are CelebA, Waterbirds, MultiNLI, and Civil Comments. There have been significant improvements on these datasets in recent years, however, often through developing increasingly complex algorithms, relying on large computational budgets, and learning multiple hyper-parameters. However, a question that arises is what are the underlying reasons for the observed improvements, and can we witness similar improvements with simpler methods? This work takes a step towards answering this question.

7.3. Contributions

We observed that common worst-group-accuracy datasets suffer from substantial class imbalances. A natural step when encountering such datasets is to rebalance the classes, a step that has been missing from the existing literature. As a result, we set out to compare state-of-the-art methods against simple balancing techniques in a systematic way. As the title of this work suggests, we arrived at the conclusion that "simple data balancing baselines" can already achieve significant improvements avoiding some of the complexities arising from other more complicated methods. This is an important and novel result, calling for a closer look at both algorithms and benchmarks.

Chapter 8

Simple data balancing achieves competitive worst-group-accuracy

8.1. Introduction

Machine learning classifiers achieve excellent test average classification accuracy when both training and testing data originate from the same distribution (Vapnik, 1995; LeCun et al., 2015). In contrast, small discrepancies between training and testing distributions cause these classifiers to fail in spectacular ways (Alcorn et al., 2019). While training and testing distributions can differ in multiple ways, we focus on the problem of *worst-group-accuracy* (Sagawa et al., 2019). In this setup, we discriminate between multiple *classes*, where each example also exhibits some (labeled or unlabeled) *attributes*. We call each class-attribute combination a *group*, and assume that the training and testing distributions differ in their group proportions. Then, our goal is to learn classifiers maximizing worst test performance across groups.

Optimizing worst-group-accuracy is relevant because it reduces the reliance of machine learning classifiers on *spurious correlations* (Arjovsky et al., 2019), that is, patterns that discriminate classes only between specific groups (Shah et al., 2020; Geirhos et al., 2018, 2020). The problem of worst-group-accuracy is also related to building fair machine learning classifiers (Barocas et al., 2019), where groups may have societal importance (Datta et al., 2014; Chouldechova, 2017; Rahmattalabi et al., 2020; Metz and Satariano, 2020).

Maximizing worst-group-accuracy is an active area of research, producing two main strands of methods (reviewed in Section 8.3). On the one hand, there are methods that consider access to attribute information during training, such as the popular group Distributionally Robust Optimization (Sagawa et al., 2019, gDRO). On the other hand, there are methods that consider access only to class information during training, such as the recently proposed Just Train Twice (Liu et al., 2021, JTT). Unsurprisingly, methods using attribute information achieve the best worst-group-accuracy. But, since labeling attributes for all examples is a costly human endeavour, alternatives such as JTT are of special interest when building machine learning systems featuring strong generalization and

requiring weak supervision.

This work takes a step back and studies the characteristics of four common datasets to benchmark worst-group-accuracy models (CelebA, Waterbirds, MultiNLI, CivilComments). In particular, we observe that these datasets exhibit a large class imbalance which, in turn, correlates with a large group imbalance (Section 8.2). In light of this observation, we study the efficacy of training systems under data subsampling or reweighting to balance classes and groups (Section 8.4).

Our experiments (Section 8.5) provide the following takeaways:

- Due to class or attribute imbalance, simple data balancing baselines achieve competitive performance in four common worst-group-accuracy benchmarks, are faster to train, and require no additional hyper-parameters.
- While we obtained the best results by balancing groups, simple class balancing is also a powerful baseline when attribute information is unavailable.
- Access to attribute information is most critical for model selection (in the validation set), and not so much during training.
- We recommend practitioners to try data subsampling first, since (i) it is faster to train, (ii) is less sensitive to regularization hyper-parameters, and (iii) has a stable performance during long training sessions.
- Given the efficacy of subsampling methods, question the mantra "just collect more data", particularly when optimizing for test *worst-group-accuracy* (as opposed to the classic goal of optimizing test *average* accuracy).

In essence, we beg for closer examination of both benchmarks and methods for future research in worst-group-accuracy optimization.

8.2. Popular worst-group-accuracy benchmarks

We consider datasets $\{(x_i, y_i, a_i)\}_{i=1}^n$, where each example is a triplet containing an input x_i, a class label y_i, and an attribute label a_i. The sequel studies four popular worst-group accuracy benchmarks that follow this structure.

- **CelebA** (Liu et al., 2015b; Sagawa et al., 2019) consists of images of aligned celebrity faces. Each face image is annotated with multiple traits. Here, our task is to classify if the person has blond hair. The attribute indicates whether the person in the image is male or female.
- **Waterbirds** (Wah et al., 2011; Sagawa et al., 2019) contains images of birds cut and pasted on different backgrounds. The task is to classify specimens into water birds or land birds. The attribute indicates whether the bird appears on its natural habitat or not.
- **MultiNLI** (Williams et al., 2017; Sagawa et al., 2019) is a dataset containing pairs of sentences. The task is to classify the relationship between the two sentences as being a contradiction, an entailment, or none of the two. The attribute indicates whether there is a

Dataset	Target	Group Counts		Class Counts	$\hat{\mathbf{P}}(\mathbf{Y}=\mathbf{y}\|\mathbf{A}=\mathbf{a})$		$\hat{\mathbf{P}}(\mathbf{Y}=\mathbf{y})$
	$\downarrow y \quad a \rightarrow$	Female	Male		Female	Male	
CelebA	Blond	22880	1387	24267	24.2%	2.0%	14.9%
	Not blond	71629	66874	138503	75.8%	98.0%	85.1%
		Water	Land		Water	Land	
Waterbirds	Land bird	56	1057	1113	1.6%	85.2%	23.2%
	Water bird	3498	184	3682	98.4%	14.8%	76.8%
		Identity	Other		Identity	Other	
CivilComments (Coarse)	Non toxic	90337	148186	238523	83.6%	92.1%	88.7%
	Toxic	17784	12731	30515	16.4%	7.9%	11.3%
		No negation	Negation		No negation	Negation	
MultiNLI	Contradiction	57498	11158	68656	30.0%	76.1%	33.3%
	Entailment	67376	1521	68897	35.2%	10.4%	33.4%
	Neutral	66630	1992	68622	34.8%	13.6%	33.3%

Table 1. Class and group counts for four popular worst-group-accuracy benchmarks. These datasets exhibit large class (y) and group imbalance. In particular, class probabilities shift significantly when conditioning on the attribute (a) value. For instance, the CelebA dataset has only 15% of examples of class "blond". Moreover, the probability of "blond" is different when the attribute value is "female" (24%) or "male" (2%), creating a spurious correlation.

negation word in the second sentence. When a negation word is present, contradiction is the most likely label.

- **CivilComments** (Borkan et al., 2019; Koh et al., 2021) is a dataset containing comments from online forums. The task is to classify whether a comment is toxic or not. There are multiple attributes annotating the content of each comment, relating to: male, female, LGBT, black, white, Christian, Muslim, other religion. Following Sagawa et al. (2019), we consider a *coarse* version of the CivilComments dataset to train gDRO.This coarse version provides a binary attribute indicating if any of the eight attributes listed above appears in the comment.

Table 1 lists the number of examples per class and group for these four datasets. The data reveals that three out of four datasets exhibit a large class imbalance, and that all of them exhibit a large group imbalance. Furthermore, these imbalances are highly correlated: class probabilities vary significantly when conditioning on the attribute value. In the Waterbirds dataset, class probabilities invert when swapping attribute values. In the MultiNLI dataset, the class "contradiction" is much more likely when there is a negation in the second sentence. In CelebA, it is unlikely to find examples from the "male" class when the attribute is "blond". Therefore, these datasets contain spurious correlations helpful to discriminate only between some groups. When such groups represent most of the dataset, learning algorithms latch onto the spurious correlations, and resort to memorization to achieve zero training error (Sagawa et al., 2020).

These observations immediately motivate training classifiers under data subsampling or reweighting to balance out classes and groups. After group balancing, we expect the spurious correlations

between classes and attributes to vanish, improving test worst-group-accuracy. Before exploring the efficacy of these simple balancing baselines, we first review some popular state-of-the-art methods proposed to optimize worst-group-accuracy.

8.3. Popular worst-group-accuracy methods

We review three popular methods in the literature of worst-group-accuracy optimization.

- **Empirical Risk Minimization** (Vapnik, 1995, **ERM**) chooses the predictor minimizing the empirical risk $\frac{1}{n}\sum_{i=1}^{n}\ell(f(x_i),y_i)$. ERM does not use attribute labels.
- **Just Train Twice** (Liu et al., 2021, **JTT**) proceeds in two steps. First, JTT trains an ERM model for a small amount of epochs T. Assuming that this "simplistic" ERM model classifies examples based on spurious correlations, its errors should correlate to the subset of examples where the spurious pattern does not appear. Following this assumption, JTT trains a final ERM model on a dataset where the mistakes from the "simplistic" ERM model appear λ_{up} times. JTT does not use attribute labels.
- **Group Distributionally Robust Optimization** (Sagawa et al., 2019, **gDRO**) minimizes the maximum loss across groups: $\sup_{q\in\Delta_{|G|}}\sum_{g=1}^{|G|}\frac{q_g}{n_g}\sum_{i=1}^{n_g}\ell(f(x_i),y_i)$, where $G=Y\times A$ is the set of all groups, $\Delta_{|G|}$ is the $|G|-$dimensional simplex and n_g is the number of examples from group $g\in G$ contained in the dataset. Therefore, gDRO uses attribute labels. In particular, gDRO allocates a dynamic weight q_g to the minimization of the empirical loss of each group, proportional to its current error.

Other methods The literature in robust optimization is flourishing, so the comparison of all possible methods renders itself impossible. Some further examples of robust learners not using attribute information are Learning from Failure (Nam et al., 2020), the Too-Good-to-be-True prior (Dagaev et al., 2021), Spectral Decoupling (Pezeshki et al., 2020), Environment Inference for Invariant Learning (Creager et al., 2021), and the GEORGE clustering algorithm (Sohoni et al., 2020). Other examples of methods that use attribute information include Conditional Value at Risk (Duchi et al., 2019), Predict then Interpolate (Bao et al., 2021), Invariant Risk Minimization (Arjovsky et al., 2019), and a plethora of domain-generalization algorithms (Gulrajani and Lopez-Paz, 2020).

8.4. Simple data balancing baselines

Given the class and group imbalance shown in Table 1, we explore the effectiveness of four data balancing baselines on worst-group-accuracy:

- Subsampling large classes (**SUBY**), so every class is the same size as the smallest class. Such subsampling is performed once and fixed before training starts. This baseline *does not use* attribute labels.

- Similarly, subsampling large groups (Sagawa et al., 2020, **SUBG**), so every group is the same size as the smallest group. This baseline *does use* attribute labels.
- Reweighting the sampling probability of each example, so mini-batches are class-balanced in expectation (**RWY**). This baseline *does not use* attribute labels.
- Similarly, reweighting the sampling probability of each example, so mini-batches are group-balanced in expectation (**RWG**). This baseline *does use* attribute labels.

To motivate our baselines, we consider a synthetic logistic regression example (Sagawa et al., 2020, Section 5.1.). The classes $y \in \{-1, +1\}$ are dependent on two attributes $a_{core}, a_{spu} \in \{-1, +1\}$ with correlations $\rho_{core}, \rho_{spu} \in [-1, 1]$. While ρ_{core} remains invariant between the training and the test data, ρ_{spu} varies from the training to the test set. Each attribute dictates a Gaussian distribution over input features. In particular, each input example x is a concatenation of the following three components,

$$x := \begin{bmatrix} \gamma_{\text{spu}} & x_{\text{spu}} \\ \gamma_{\text{core}} & x_{\text{core}} \\ \gamma_{\text{noise}} & x_{\text{noise}} \end{bmatrix} \in \mathbb{R}^{d+2}, \qquad \text{where} \qquad \begin{aligned} \mathbb{P}(x_{spu} \mid y) &= \mathcal{N}(a_{spu}, \sigma^2) \in \mathbb{R}, \\ \mathbb{P}(x_{core} \mid y) &= \mathcal{N}(a_{core}, \sigma^2) \in \mathbb{R}, \\ \mathbb{P}(x_{noise} \mid y) &= \mathcal{N}(0\,, \sigma^2) \in \mathbb{R}^d, \end{aligned} \tag{8.4.1}$$

where (σ^2, γ_i) are the variance and scale of each of the features. The scaling factors γ_i control the rate at which the model learns each feature: the larger γ_i, the faster the model learns about x_i. Moreover, the noise features x_{noise} are independent from the class labels y, and therefore uncorrelated *in expectation*. However, in over-parameterized settings where d is greater than the number of training examples, there exists an *empirical* correlation between the noise features and the class labels. Therefore, and depending on the values of γ_i, over-parametrized models can exploit noise features to memorize training examples on their path to achieving zero training error.

Figure 1 implements one instance of this example where $\rho_{\text{core}} = 1$ and $\rho_{\text{spu}} = 0.8$. Furthermore, $\gamma_{\text{spu}} = 4$, $\gamma_{\text{core}} = 1$, $\gamma_{\text{noise}} = 20$, and $\sigma = 0.15$. Given these correlation and scaling coefficients, the spurious feature is learnable much faster than the core and noise features. As shown on the first two panels of Figure 1, a vanilla ERM model mainly relies on spurious features, and therefore achieves poor test worst-group-accuracy. On the other hand, subsampling the majority group (SUBG) decorrelates the spurious feature from the labels, leading to a model that relies on the core feature, discards the spurious feature, and achieves good test worst-group-accuracy. While reweighting groups (RWG) also solves this toy example, one has to pay special attention to model selection, since test worst-group-accuracy degrades as the number of training iterations increases. We note that the probability of misclassifying when using just x_{core} is $P(yx_{\text{core}} < 0) = P(y(1 + y\sigma Z) < 0) = P\left(Z < -\frac{y}{\sigma}\right) = 1 - \Phi\left(-\frac{y}{\sigma}\right) = 1 - \Phi\left(\frac{1}{\sigma}\right)$ (since Φ is symmetrical) which is 10^{-11} in this particular case. This means that the problem is separable using just x_{core} with high probability.

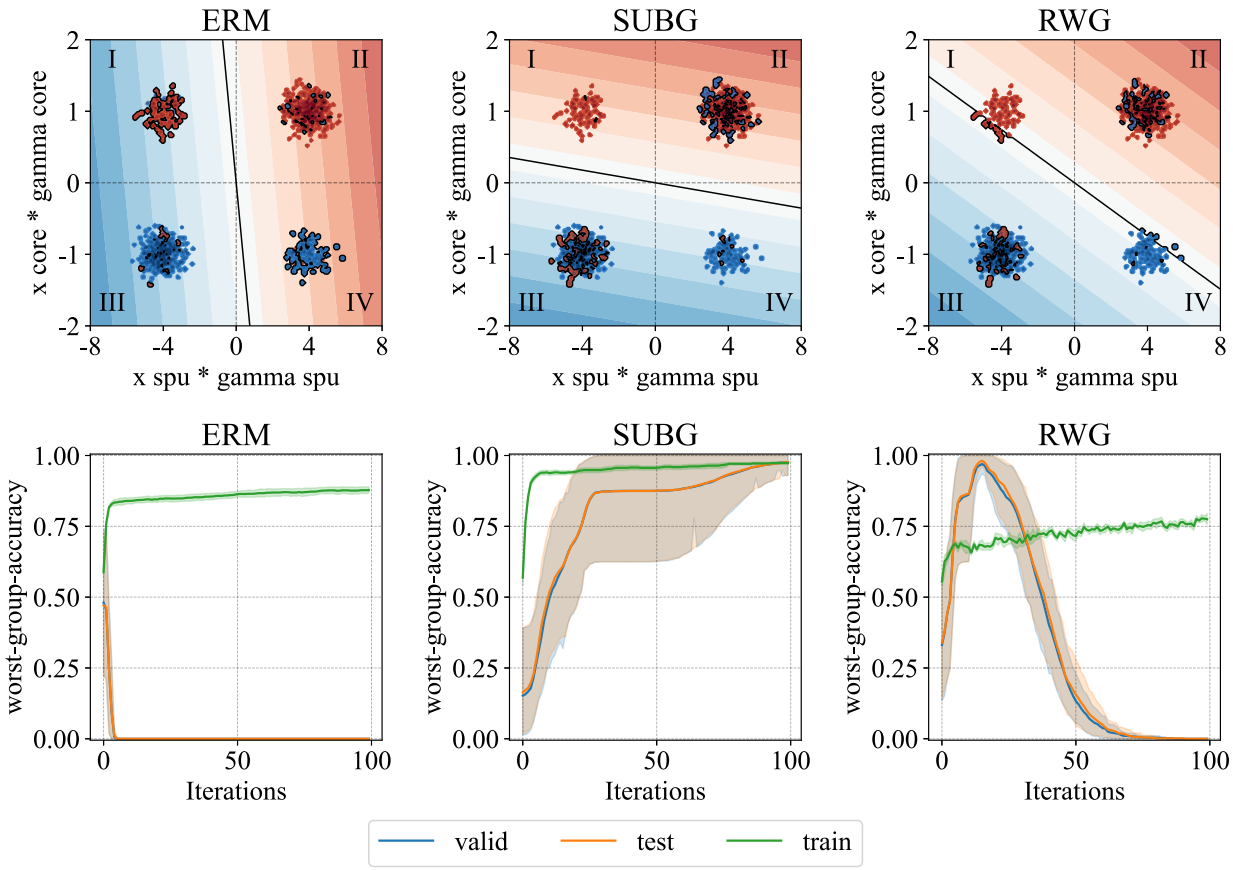


Fig. 1. A linear binary classification task with a spurious feature (x-axis, ranging from -8 to 8), a core feature (y-axis, ranging from -2 to 2), and 1200 noise features (not depicted, Normally distributed). Each class contains a majority group (quadrants II and III) and a minority group (quadrants I and IV). Shades of red show the predicted probability of class +1 and shades of blue the predicted probability of class -1, under the classifier w. The value of each position x on the heatmap is sigmoid$(w^T \tilde{x})$, where we get $\tilde{x}$ by adding the noise vector of the sample in the training set closest to x. This enables us to visualize the regions of the 2D space $(x_{\text{spu}}, x_{\text{core}})$ where the model uses the noise vectors to predict. We depict the performance of three models at their best iteration with respect to validation worst-group-accuracy. **On the left**, an ERM model finds the easy solution of using (i) the spurious feature to discriminate between majority examples of each class, and (ii) the noise features to memorize the minority examples of each class (shown as small neighbourhoods). This leads to poor test worst-group-accuracy. **On the middle**, subsampling the majority groups (SUBG) of each class decorrelates the spurious feature and the class label, guiding the model to rely on the core feature. This leads to improved test worst-group-accuracy. **On the right**, balancing groups by data reweighting (RWG) also achieves good test worst-group-accuracy, but only when early-stopping the training process carefully. The figures are averages over eight random seeds.

8.5. Experiments

We implement ERM, JTT, gDRO, SUBY, SUBG, RWY and RWG, as well as the necessary infrastructure to experiment on the Waterbirds, CelebA, MultiNLI, and CivilComments benchmarks. Our implementation follows closely the ones of (Sagawa et al., 2019, gDRO) and (Liu et al., 2021, JTT).

Method	#HP	Groups	Worst Acc				Average
			CelebA	Waterbirds	MultiNLI	CivilComments	
ERM	4	No	79.7±3.7	85.5±1.0	67.6±1.2	61.3±2.0	73.5
JTT	6	No	75.6±7.7	85.6±0.2	67.5±1.9	67.8±1.6	74.1
RWY	4	No	82.9±2.2	86.1±0.7	68.0±1.9	67.5±0.6	76.2
SUBY	4	No	79.9±3.3	82.4±1.7	64.9±1.4	51.2±3.0	69.6
RWG	4	Yes	84.3±1.8	87.6±1.6	69.6±1.0	72.0±1.9	78.4
SUBG	4	Yes	85.6±2.3	89.1±1.1	68.9±0.8	71.8±1.4	78.8
gDRO	5	Yes	86.9±1.1	87.1±3.4	78.0±0.7	69.9±1.2	80.5

Table 2. Averages and standard deviations of test worst-group-accuracies for all methods and datasets. #HP is the number of tuned hyper-parameters. Simple data balancing baselines match the performance of state-of-the-art methods within error bars, with two exceptions. Green backgrounds indicate datasets where algorithms exhibit a statistically different performance at a significance level of $\alpha = 0.05$. This is determined using an Alexander-Govern test for the equality of means of multiple sets of samples with heterogeneous variance (Alexander and Govern, 1994). All algorithms not using attribute information perform similarly with the exception of SUBY, under-performing in CivilComments. All algorithms using attribute information perform similarly, with the exception of gDRO being better on MultiNLI.

For the image datasets Waterbirds and CelebA, we train ResNet50 models pre-trained on ImageNET (He et al., 2016) using the SGD optimizer. For the NLP datasets MultiNLI and CivilComments, we train BERT models pre-trained on Book Corpus and English Wikipedia (Devlin et al., 2018) using the AdamW optimizer (Loshchilov and Hutter, 2017). We tune the learning rate in $\{10^{-5}, 10^{-4}, 10^{-3}\}$, weight decay in $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$, and JTT's λ_{up} in $\{4, 5, 6, 20, 50, 100\}$. We tune the batch size in $\{2, 4, 8, 16, 32, 64, 128\}$ for CelebA and Waterbirds and $\{2, 4, 8, 16, 32\}$ for MultiNLI and CivilComments. We tune JTT's T in $\{40, 50, 60\}$ for Waterbirds, $\{1, 5, 10\}$ for CelebA, and $\{1, 2\}$ for MultiNLI and CivilComments. We fix gDRO's η to 0.1 We allow 50 random combinations of hyper-parameters for each method and dataset. In contrast to previous literature, we run each hyper-parameter random combination 5 times to compute the average and standard deviation of the reported test worst-group-accuracies. These error-bars relate to data shuffling, data subsampling, and random initialization of last linear layers. We train Waterbirds for 360 epochs, CelebA for 60 epochs, and both MultiNLI and CivilComments for 7 epochs. We select best models (hyper-parameter combination and epoch) by computing the worst-group-accuracy on a validation set. Our code is available at `https://github.com/facebookresearch/BalancingGroups`.

	ERM	JTT	RWY	SUBY	RWG	SUBG	gDRO
Best test worst-group-accuracy	73.5	74.1	76.2	69.6	78.4	78.8	80.5
↳ w/o attributes in validation set	-15.6	-19.7	-13.1	-24.4	-17.2	-10.4	-14.9
↳ w/o regularization	-9.4	-9.9	-6.5	-8.5	-12	-1.5	-10.3
Minutes per epoch	39	74	39	19	33	5	39

Table 3. Some ablations on the experimental results, averaged over datasets. The **first row** shows the best results test worst-group-accuracy, averaged across datasets, obtained by employing a validation set with attribute annotations and allowing model regularization. The **second row** shows the drop in test worst-group-accuracy when performing model selection based on *average* validation accuracy (no attribute annotations). The **third row** shows the drop in test worst-group-accuracy when performing model selection only amongst those models with weak regularization (no early stopping, weight-decay 10^{-4}). The **fourth row** shows median running time per epoch (in minutes). SUBG is the only algorithm whose performance does not degrade when taking out regularization.

8.5.1. Results

Table 2 reports test worst-group-accuracies for all methods and benchmarks. While some methods do not require the use of attribute labels for training, we emphasize that *all methods require a validation set with attribute labels* to perform model selection. As shown in the second row of Table 3, the performance of all methods degrades when one performs model selection based on the average validation accuracy (e.g., not assuming access to attribute labels in the validation set). Table 2 also lists the number of hyper-parameters tuned by each method, four being the minimal achieved by ERM, SUBG, SUBY, RWG, RWY (learning rate, weight decay, batch size, early stopping epoch). In summary, reweighting baselines perform competitively: SUBG scores only 1.7 points less than gDRO on average, while RWY scores 2.1 points more than JTT on average. Subsampling SUBY performs below its reweighting counterpart RWY, while SUBG outperforms RWG by a small margin. Finally, the fourth row of Table 3 reports the running times employed to find the best models discussed above. This shows that ERM and RWY is 1.9 times faster than its competitor JTT, and that RWG is 1.2 times faster than its competitor gDRO.The subsampling baselines are 3.8 times faster than JTT and 7 times faster than gDRO while only having slightly worse worst-group-accuracy.

8.5.2. Hyper-parameter analysis

Table 4 summarizes the top 5 best hyper-parameters for each dataset and method, together with their associated test worst-group-accuracies. We make three observations. First, the range of test worst-group-accuracies (top 1st worst-group-accuracy - top 5th worst-group-accuracy) is smaller for methods accessing group information. This means that if we used less than 50 hyperparameter tuning runs, we would still get a good worst-group-accuracy, which implies that these methods

Dataset	Groups	Method	Hyperparameters				Worst Acc	
			$\log_{10}$(LR)	$\log_{10}$(WD)	Epoch	Batch Size	Range	Δ
CelebA	No	ERM	-3.4±0.5	-1.0±0.0	37.1±5.0	128.0±0.0	[75.4, 80.8]	5.4
		JTT	-3.4±0.9	-1.8±0.4	30.8±6.4	48.0±50.3	[70.6, 76.3]	5.8
		RWY	-4.6±0.5	-1.4±0.5	14.0±8.7	2.8±1.1	[78.9, 82.9]	4.1
		SUBY	-4.4±0.9	-1.2±0.4	31.4±14.4	42.0±54.3	[78.4, 79.9]	1.4
	Yes	RWG	-5.0±0.0	-1.0±0.0	6.0±4.1	36.8±26.3	[82.8, 84.4]	1.7
		SUBG	-4.2±0.4	-2.0±1.2	27.0±11.5	4.8±1.8	[83.9, 86.6]	2.7
		gDRO	-5.0±0.0	-3.2±1.3	15.4±0.7	64.0±0.0	[86.7, 87.4]	0.8
Civil Comments	No	ERM	-3.8±0.4	-3.6±0.5	3.6±0.9	6.8±5.6	[60.4, 61.3]	0.9
		JTT	-5.0±0.0	-1.8±1.5	4.5±0.4	25.6±8.8	[62.6, 68.3]	5.7
		RWY	-3.6±0.5	-3.6±0.5	4.3±0.8	10.8±12.1	[52.6, 68.3]	15.7
		SUBY	-3.4±0.9	-3.4±0.9	3.1±0.6	25.6±8.8	[49.2, 51.2]	2.1
	Yes	RWG	-4.8±0.4	-2.4±0.5	2.5±0.4	6.4±5.4	[71.4, 72.0]	0.6
		SUBG	-3.2±0.4	-4.0±0.0	3.5±0.5	17.6±8.8	[70.2, 71.8]	1.6
		gDRO	-3.2±0.4	-3.4±0.5	3.5±1.6	26.4±12.5	[68.0, 69.9]	1.9
MultiNLI	No	ERM	-3.8±0.4	-4.0±0.0	4.6±0.4	8.4±13.2	[65.6, 67.6]	2.0
		JTT	-5.0±0.0	-2.2±0.8	4.3±0.9	4.4±2.2	[65.3, 67.5]	2.1
		RWY	-3.8±0.4	-3.8±0.4	4.1±0.9	9.2±6.6	[60.0, 68.0]	8.0
		SUBY	-3.8±0.8	-3.0±0.0	3.8±0.4	9.2±6.6	[56.2, 64.9]	8.7
	Yes	RWG	-4.8±0.4	-2.4±0.5	2.2±0.4	9.2±12.8	[68.1, 69.8]	1.7
		SUBG	-3.4±0.5	-3.2±0.8	5.3±0.3	13.6±12.4	[68.5, 68.9]	0.4
		gDRO	-4.0±0.0	-3.4±0.5	5.1±0.8	19.2±12.1	[76.4, 78.0]	1.5
Waterbirds	No	ERM	-4.2±0.4	-2.8±1.3	207.6±107.5	4.0±2.4	[79.4, 85.6]	6.2
		JTT	-3.6±0.5	-2.8±1.1	187.9±117.6	3.6±0.9	[83.7, 85.6]	2.0
		RWY	-4.4±0.5	-2.2±0.8	129.6±84.4	4.0±2.4	[83.2, 86.1]	2.9
		SUBY	-4.8±0.4	-3.4±0.9	238.6±47.6	2.0±0.0	[79.2, 82.4]	3.2
	Yes	RWG	-5.0±0.0	-1.0±1.0	88.6±100.9	36.0±52.9	[85.4, 87.6]	2.2
		SUBG	-4.0±0.0	-2.6±1.1	192.0±44.7	4.8±1.8	[87.9, 89.1]	1.1
		gDRO	-5.0±0.0	-0.6±0.5	23.4±27.2	4.0±2.4	[86.4, 88.2]	1.8

Table 4. Means and standard deviations of the hyper-parameters chosen by the top 5 runs for each dataset and method. The last column shows the range of the associated test worst-group-accuracies. Blue indicates low values, yellow indicates large values.

are less sensitive to hyper-parameter choice. Second, methods are most sensitive to the choice of learning rate, with multiple sets of top-5 runs preferring the same value. Third, RWG prefers small-capacity models by choosing small learning rates, high weight decays, and early epochs.

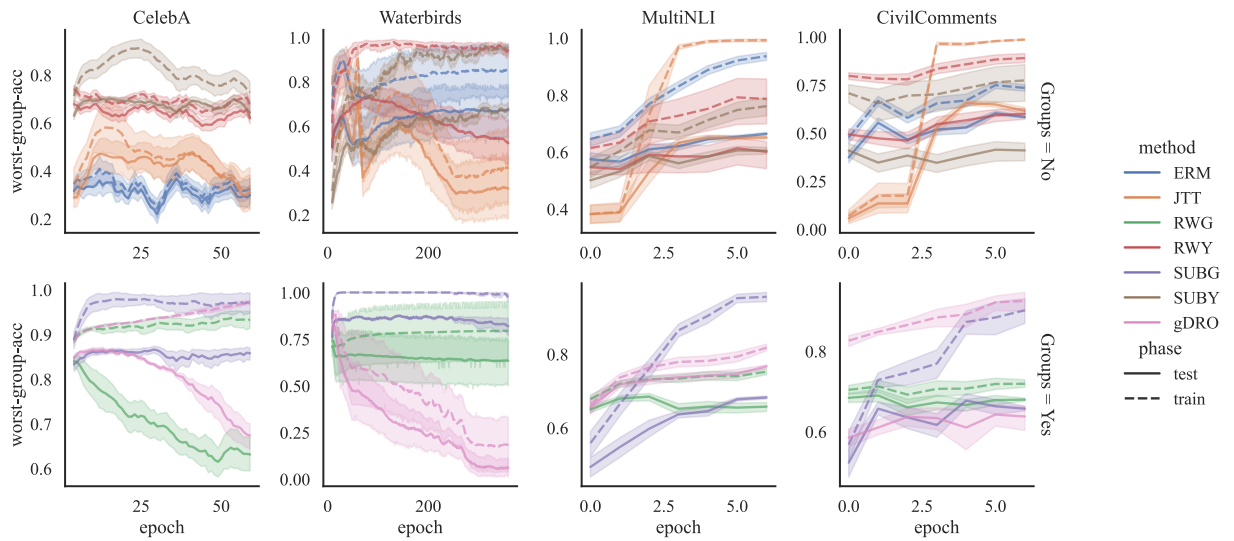


Fig. 2. Average evolution of worst-group-accuracy for the top-5 best runs of each dataset and method. While reweighting methods (RWY, RWG, gDRO, JTT, ERM) sometimes degrade in performance over long training sessions, subsampling methods (SUBY, SUBG) show a more robust behavior.

8.5.3. Evolution of worst-group-accuracy during training

Figure 2 shows the evolution of the train and test worst-group-accuracy for all methods and datasets. First, we observe that RWY, RWG, and gDRO peak in worst-group-accuracy early, and then degrade in performance. On the contrary, SUBG has a more stable performance during long sessions of training, for all datasets and especially in Waterbirds. Second, there is a consistent generalization gap for all methods and datasets regardless of regularization strength. Since some models reach 100% *train* worst-group-accuracy, they must have memorized some of the worst-group examples.

8.5.4. Differences between reweighting and subsampling groups

While similar at a first glance, training models with data subsampling or reweighting may lead to different decision boundaries due to different interactions with regularization (Sagawa et al., 2020). For instance, in the absence of regularization, logistic regression converges to the maximum margin classifier (Soudry et al., 2018b) in linear realizable problems. Therefore, since any strictly positive reweighting of example probabilities *has no effect on support vectors*, we conclude that regularization is necessary for reweighting to have any effect on the resulting classifier. In contrast, subsampling changes the support of the dataset, likely removing support vectors and affecting the final classifier *even in the absence of regularization*.

While the previous are proven facts only for linear problems, Table 3 shows similar findings for the over-parametrized deep models used in this work. In particular, the reweighting methods (RWG

and RWY) and gDRO, both degrade when removing regularization. That is consistent with findings of Słowik and Bottou (2021) where they establish close theoretical connections between gDRO and reweighting mechanisms. On the other hand, the subsampling method (SUBG) maintains its performance in the long run without the need of regularization. Byrd and Lipton (2019); Sagawa et al. (2019) reach a similar conclusion: strong regularization is necessary to benefit from data reweighting. Table 4 ratifies this, since SUBG prefers smaller weight decays than RWG.The superior performance of SUBG suggests two conclusions. On the one hand, we favor subsampling under tight computational budgets, since the resulting models are faster to train and depend less on regularization hyper-parameters. On the other hand, the considered benchmarks seem solvable with small data, showing that *either the tasks at hand are too easy or that the reweighting methods fail to make good use of all the data*.

To conclude, we comment on one similarity between early-stopped reweighting and subsampling. Reweighting uses a weighted random sampler to produce mini-batches containing an equal amount of minority and majority examples (in expectation). This weighted random sampler is with replacement due to the scarcity of minority examples. Therefore, for a small amount of epochs, the model has likely seen all the minority examples while only observed a *subsample* of the majority examples. More specifically, the number of observed unique majority examples after sampling k times is on average $N_{\text{maj}}(1-(1-\frac{1}{N_{\text{maj}}})^k)$, where N_{maj} is the number of majority examples contained in the dataset. Given that the best RWG model stops after 3 epochs for CelebA, it observes only 44% of majority examples on average, which amounts to subsampling the majority group.

8.6. Conclusion

We have shown that simple data balancing baselines achieve state-of-the-art performance in four popular worst-group-accuracy benchmarks. While balancing groups leads to best worst-group-accuracy, balancing class labels obtains competitive performance even in the absence of attribute information. We have also revisited the critical importance of having access to attribute information in the validation set, necessary to perform model selection based on worst-group-accuracy. Therefore, hyper-parameter tuning for domain generalization under weak supervision remains an open problem (Gulrajani and Lopez-Paz, 2020). We have illustrated some differences between data reweighting and data subsampling, advocating to try data subsampling first, since (i) it is faster to train and thus allows more hyper-parameter exploration, (ii) has less reliance on regularization, and (iii) has a more stable performance during long training sessions. All in all, our results raise two questions. First, are our current worst-group-accuracy benchmarks expressing a real problem? If so, is there room to outperform simple data balancing baselines in these datasets?

Chapter 9

General conclusion

Understanding the inner mechanics of neural networks, in some ways, is comparable to understanding the human brain. Some of the significant advances in neuroscience have come from observing patients with brain pathologies and puzzling out the underlying reasons. Similarly, the failure modes of neural networks and their puzzling behaviors serve as guiding signals towards finding a better understanding of neural networks. To that end, in this dissertation, we presented three articles in which we studied three particular generalization behaviors of neural networks:

- In the first article, we studied the epoch-wise double descent phenomenon. We leveraged tools from statistical physics to study a simple teacher-student setup exhibiting epoch-wise double descent similar to deep neural networks. We derived closed-form analytical expressions for the evolution of generalization error as a function of the training time. We provided a new mechanistic explanation of epoch-wise double descent and validated our findings through simple numerical experiments where our theory accurately predicts empirical findings.
- In the second article, we identified and formalized gradient starvation (GS), a deficiency of gradient descent when optimizing the cross-entropy loss. Using tools from dynamical systems we showed that GS could slow down the learning of certain features leading to adverse but also possibly beneficial consequences. If the learned features are sufficient to generalize to the test data, gradient starvation can be viewed as an implicit regularizer. Otherwise, gradient starvation could have an unfavorable effect, which we observed empirically when some predictive features fail to be learned.
- In the third article, we studied the problem of generalization to underrepresented groups. Particularly, we studied datasets and models which achieve good generalization performance on average but perform poorly on minority groups of examples. Our results suggest that simple data balancing methods achieve state-of-the-art accuracy on minority groups, calling for closer examination of benchmarks and methods for research in out-of-distribution generalization.

These findings will hopefully facilitate further progress in understanding the dynamics of neural networks. Particularly, this dissertation highlighted two important aspects of research in neural

networks' theory:

- **The significance of incorporating the structure of data into neural networks' analysis** was particularly highlighted in the first and second articles. We observed the crucial role of having multi-scale features which were shown to result in the double descent phenomenon. Developing a solid theory of learning in neural networks necessitates incorporating the interplay between the structure of data and other components of learning.
- **The importance and benefits of studying simpler models** were underlined in the first and third articles. In fact, with a plethora of increasingly complex models and algorithms, it is becoming more and more difficult to get insight into the inner mechanics of learning. However, as our findings suggest, interesting behaviors of complex neural networks can already be observed in simpler linear models. Additionally, the third article highlighted the potential of incredibly simpler models that require far less tuning and yet achieve competitive performances and are easier to analyze. These findings offer the community a solid reason to consider simpler models/algorithms when studying a phenomenon or tackling a problem.

That being said, it is also important to stress that there are several limitations to our studies on learning theory. For example, in our analysis, we took into account the strength of different features but imposed strong conditions on their distribution. Real world data such as images have much more complex structure which renders similar analyses infeasible or extremely non-trivial. Another limitation of our analysis is the simplicity of toy models. As much as toy models are important for development of tractable analysis, they may fail to display many behaviors of large-scale neural networks – behaviors that arise from entangled interactions between different elements of learning. Certain behaviors are emergent only at a certain scale of complexity.

Despite the long way ahead towards developing a general and unified theory of learning, we hope that the application of scientific experiments can set the stage for further theoretical analysis and will enable the researchers to see through the haze of empirical results that surround us.

References

Advani, M. S. and Saxe, A. M. (2017a). High-dimensional dynamics of generalization error in neural networks. *CoRR*, abs/1710.03667.

Advani, M. S. and Saxe, A. M. (2017b). High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*.

Advani, M. S., Saxe, A. M., and Sompolinsky, H. (2020). High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446.

Ahuja, K., Shanmugam, K., and Dhurandhar, A. (2020a). Linear regression games: Convergence guarantees to approximate out-of-distribution solutions. *arXiv preprint arXiv:2010.15234*.

Ahuja, K., Shanmugam, K., Varshney, K., and Dhurandhar, A. (2020b). Invariant risk minimization games. *arXiv preprint arXiv:2002.04692*.

Akhtar, N. and Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430.

Albert, A. and Anderson, J. A. (1984). On the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, 71(1):1–10.

Alcorn, M. A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.-S., and Nguyen, A. (2019). Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects.

Alexander, R. A. and Govern, D. M. (1994). A New and Simpler Approximation for ANOVA under Variance Heterogeneity. 19(2):91–101.

Ali, A., Dobriban, E., and Tibshirani, R. (2020). The implicit regularization of stochastic gradient flow for least squares. In *International Conference on Machine Learning*, pages 233–244. PMLR.

Ali, A., Kolter, J. Z., and Tibshirani, R. J. (2019). A continuous-time view of early stopping for least squares regression. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1370–1378. PMLR.

Allen-Zhu, Z., Li, Y., and Liang, Y. (2019a). Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6158–6169.

Allen-Zhu, Z., Li, Y., and Song, Z. (2019b). A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR.

Amari, S.-i., Ba, J., Grosse, R., Li, X., Nitanda, A., Suzuki, T., Wu, D., and Xu, J. (2020). When does preconditioning help or hurt generalization? *arXiv preprint arXiv:2006.10732*.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. (2019). Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Arora, S., Cohen, N., Hu, W., and Luo, Y. (2019a). Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pages 7413–7424.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. (2019b). On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pages 8139–8148.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. (2019c). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*.

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. *arXiv preprint arXiv:1706.05394*.

Ba, J., Erdogdu, M., Suzuki, T., Wu, D., and Zhang, T. (2019). Generalization of two-layer neural networks: An asymptotic viewpoint. In *International conference on learning representations*.

Bai, Y. and Lee, J. D. (2020). Beyond linearization: On quadratic and higher-order approximation of wide neural networks.

Baker, N., Lu, H., Erlikhman, G., and Kellman, P. J. (2018). Deep convolutional networks do not classify based on global object shape. *PLoS computational biology*, 14(12):e1006613.

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

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

Bao, Y., Chang, S., and Barzilay, R. (2021). Predict then interpolate: A simple algorithm to learn stable classifiers. *arXiv preprint arXiv:2105.12628*.

Baratin, A., George, T., Laurent, C., Hjelm, R. D., Lajoie, G., Vincent, P., and Lacoste-Julien, S. (2020). Implicit regularization in deep learning: A view from function space. *arXiv preprint arXiv:2008.00938*.

Barocas, S., Hardt, M., and Narayanan, A. (2019). *Fairness and Machine Learning*. fairmlbook.org. http://www.fairmlbook.org.

Bartlett, P. (1998). The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, 44(2):525–536.

Beery, S., Van Horn, G., and Perona, P. (2018). Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473.

Belinkov, Y. and Bisk, Y. (2017). Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019a). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019b). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854.

Belkin, M., Hsu, D., and Xu, J. (2020). Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180.

Bender, C. M. and Orszag, S. A. (2013). *Advanced mathematical methods for scientists and engineers I: Asymptotic methods and perturbation theory*. Springer Science & Business Media.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer.

Bietti, A. and Mairal, J. (2019). On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems*, pages 12893–12904.

Borkan, D., Dixon, L., Sorensen, J., Thain, N., and Vasserman, L. (2019). Nuanced metrics for measuring unintended bias with real data for text classification. *WWW*.

Bös, S. (1998). Statistical mechanics approach to early stopping and weight decay. *Physical Review E*, 58(1):833.

Bös, S., Kinzel, W., and Opper, M. (1993). Generalization ability of perceptrons with continuous outputs. *Physical Review E*, 47(2):1384.

Bottou, L. et al. (1991). Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12.

Breiman, L. (2018). Reflections after refereeing papers for nips. In *The Mathematics of Generalization*, pages 11–15. CRC Press.

Brendel, W. and Bethge, M. (2019). Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *arXiv preprint arXiv:1904.00760*.

Brutzkus, A., Globerson, A., Malach, E., and Shalev-Shwartz, S. (2017). Sgd learns over-parameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174*.

Burges, C. J. and Crisp, D. J. (2000). Uniqueness of the svm solution. *Advances in neural information processing systems*, 12:223–229.

Byrd, J. and Lipton, Z. C. (2019). What is the effect of importance weighting in deep learning? In *ICML*.

Cao, Y., Fang, Z., Wu, Y., Zhou, D.-X., and Gu, Q. (2019). Towards understanding the spectral bias of deep learning. *arXiv preprint arXiv:1912.01198*.

Cao, Y. and Gu, Q. (2019). Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, pages 10836–10846.

Chen, L., Min, Y., Belkin, M., and Karbasi, A. (2020a). Multiple descent: Design your own generalization curve. *arXiv preprint arXiv:2008.01036*.

Chen, Z., Cao, Y., Gu, Q., and Zhang, T. (2020b). A generalized neural tangent kernel analysis for two-layer neural networks. *arXiv preprint arXiv:2002.04026*.

Chizat, L. and Bach, F. (2018). A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*, 1.

Chizat, L. and Bach, F. (2020). Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338. PMLR.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2014). The loss surface of multilayer networks. *CoRR*, abs/1412.0233.

Chouldechova, A. (2017). Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

Combes, R. T. d., Pezeshki, M., Shabanian, S., Courville, A., and Bengio, Y. (2018). On the learning dynamics of deep neural networks. *arXiv preprint arXiv:1809.06848*.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, EC-14(3):326–334.

Creager, E., Jacobsen, J.-H., and Zemel, R. (2021). Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR.

Dagaev, N., Roads, B. D., Luo, X., Barry, D. N., Patil, K. R., and Love, B. C. (2021). A too-good-to-be-true prior to reduce shortcut reliance. *arXiv preprint arXiv:2102.06406*.

d'Ascoli, S., Gabrié, M., Sagun, L., and Biroli, G. (2021). On the interplay between data structure and loss function in classification problems. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

d'Ascoli, S., Sagun, L., and Biroli, G. (2020). Triple descent and the two kinds of overfitting: Where & why do they appear? *arXiv preprint arXiv:2006.03509*.

Datta, A., Tschantz, M. C., and Datta, A. (2014). Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. *arXiv preprint arXiv:1408.6491*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Du, S. S., Hu, W., and Lee, J. D. (2018a). Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In *Advances in Neural Information Processing Systems*, pages 384–395.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. (2018b). Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*.

Duchi, J. C., Hashimoto, T., and Namkoong, H. (2019). Distributionally robust losses against mixture covariate shifts. *Under review*.

d'Ascoli, S., Refinetti, M., Biroli, G., and Krzakala, F. (2020). Double trouble in double descent: Bias and variance (s) in the lazy regime. In *International Conference on Machine Learning*, pages 2280–2290. PMLR.

Edwards, S. F. and Anderson, P. W. (1975). Theory of spin glasses. *Journal of Physics F: Metal Physics*, 5(5):965.

Engel, A. and Van den Broeck, C. (2001). *Statistical mechanics of learning*. Cambridge University Press.

Forsythe, G. E. and Moler, C. B. (1967). Computer solution of linear algebraic systems.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Galstyan, T., Harutyunyan, H., Khachatrian, H., Steeg, G. V., and Galstyan, A. (2021). Failure modes of domain generalization algorithms. *arXiv preprint arXiv:2111.13733*.

Gardner, E. (1988). The space of interactions in neural network models. *Journal of physics A: Mathematical and general*, 21(1):257.

Gardner, E. and Derrida, B. (1988). Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and general*, 21(1):271.

Gardner, E. and Derrida, B. (1989). Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983.

Geiger, M., Jacot, A., Spigler, S., Gabriel, F., Sagun, L., d'Ascoli, S., Biroli, G., Hongler, C., and Wyart, M. (2020). Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(2):023401.

Geiger, M., Spigler, S., d'Ascoli, S., Sagun, L., Baity-Jesi, M., Biroli, G., and Wyart, M. (2019). Jamming transition as a paradigm to understand the loss landscape of deep neural networks. *Physical Review E*, 100(1):012115.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *arXiv preprint arXiv:2004.07780*.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. (2018). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58.

George, T. (2020). Nngeometry: Easy and fast fisher information matrices and neural tangent kernels in pytorch. *0*.

Gerace, F., Loureiro, B., Krzakala, F., Mézard, M., and Zdeborová, L. (2020). Generalisation error in learning with random features and the hidden manifold model. In *International Conference on Machine Learning*, pages 3452–3462. PMLR.

Gidel, G., Bach, F., and Lacoste-Julien, S. (2019). Implicit regularization of discrete gradient dynamics in linear neural networks. In *Advances in Neural Information Processing Systems*, pages 3202–3211.

Goldt, S., Advani, M., Saxe, A. M., Krzakala, F., and Zdeborová, L. (2019). Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. In *Advances in Neural Information Processing Systems*, pages 6981–6991.

Goldt, S., Reeves, G., Mézard, M., Krzakala, F., and Zdeborová, L. (2020). The gaussian equivalence of generative models for learning with two-layer neural networks. *arXiv e-prints*, pages arXiv–2006.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*. MIT press Cambridge.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gulrajani, I. and Lopez-Paz, D. (2020). In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*.

Gunasekar, S., Lee, J. D., Soudry, D., and Srebro, N. (2018). Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pages 9461–9471.

Gunasekar, S., Woodworth, B. E., Bhojanapalli, S., Neyshabur, B., and Srebro, N. (2017). Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pages 6151–6159.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S. R., and Smith, N. A. (2018). Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. (2019). Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Heckel, R. and Yilmaz, F. F. (2020). Early stopping in deep networks: Double descent and how to eliminate it. *arXiv preprint arXiv:2007.10099*.

Heinze-Deml, C. and Meinshausen, N. (2017). Conditional variance penalties and domain shift robustness. *arXiv preprint arXiv:1710.11469*.

Hendrycks, D. and Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

Hendrycks, D. and Gimpel, K. (2016). A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Hermann, K. L. and Lampinen, A. K. (2020). What shapes feature representations? exploring datasets, architectures, and training. *arXiv preprint arXiv:2006.12433*.

Heskes, T. M. and Kappen, B. (1993a). On-line learning processes in artificial neural networks. In *North-Holland Mathematical Library*, volume 51, pages 199–233. Elsevier.

Heskes, T. M. and Kappen, B. (1993b). On-line learning processes in artificial neural networks.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S. S., and Sundararajan, S. (2008). A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th international conference on Machine learning*, pages 408–415.

Huang, J. and Yau, H.-T. (2019). Dynamics of deep neural networks and neural tangent hierarchy. *arXiv preprint arXiv:1909.08156*.

Huang, K., Wang, Y., Tao, M., and Zhao, T. (2020). Why do deep residual networks generalize better than deep feedforward networks?—a neural tangent kernel perspective. *Advances in Neural Information Processing Systems*, 33.

Hui, L. and Belkin, M. (2020). Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. *arXiv preprint arXiv:2006.07322*.

Idrissi, B. Y., Arjovsky, M., Pezeshki, M., and Lopez-Paz, D. (2021). Simple data balancing achieves competitive worst-group-accuracy. *arXiv preprint arXiv:2110.14503*.

Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. (2018). Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Jaakkola, T. S. and Haussler, D. (1999). Probabilistic kernel regression models. In *AISTATS*.

Jacobsen, J.-H., Behrmann, J., Zemel, R., and Bethge, M. (2018). Excessive invariance causes adversarial vulnerability. *arXiv preprint arXiv:1811.00401*.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580.

Jacot, A., Simsek, B., Spadaro, F., Hongler, C., and Gabriel, F. (2020). Implicit regularization of random feature models. In *International Conference on Machine Learning*, pages 4631–4640. PMLR.

Ji, Z. and Telgarsky, M. (2019). The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798.

Jo, J. and Bengio, Y. (2017). Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*.

Jolicoeur-Martineau, A. and Mitliagkas, I. (2019). Connections between support vector machines, wasserstein distance and gradient-penalty gans. *arXiv preprint arXiv:1910.06922*.

Kabashima, Y., Wadayama, T., and Tanaka, T. (2009). A typical reconstruction limit for compressed sensing based on lp-norm minimization. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(09):L09003.

Kalimeris, D., Kaplun, G., Nakkiran, P., Edelman, B., Yang, T., Barak, B., and Zhang, H. (2019). Sgd on neural networks learns functions of increasing complexity. *Advances in neural information processing systems*, 32.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koh, P. W., Sagawa, S., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., et al. (2021). Wilds: A benchmark of in-the-wild distribution shifts. *ICML*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Krogh, A. and Hertz, J. A. (1992a). Generalization in a linear perceptron in the presence of noise. *Journal of Physics A: Mathematical and General*, 25(5):1135.

Krogh, A. and Hertz, J. A. (1992b). A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Priol, R. L., and Courville, A. (2020). Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688*.

Krzakala, F., Mézard, M., Sausset, F., Sun, Y., and Zdeborová, L. (2012). Statistical-physics-based reconstruction in compressed sensing. *Physical Review X*, 2(2):021005.

Kuhn, R. and Bos, S. (1993). Statistical mechanics for neural networks with continuous-time dynamics. *Journal of Physics A: Mathematical and General*, 26(4):831.

Kunin, D., Sagastuy-Brena, J., Ganguli, S., Yamins, D. L., and Tanaka, H. (2020). Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. *arXiv preprint arXiv:2012.04728*.

Lampinen, A. K. and Ganguli, S. (2018). An analytic theory of generalization dynamics and transfer learning in deep linear networks. *arXiv preprint arXiv:1809.10374*.

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K.-R. (2019). Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1–8.

Le Cun, Y., Kanter, I., and Solla, S. A. (1991). Eigenvalues of covariance matrices: Application to neural-network learning. *Physical Review Letters*, 66(18):2396.

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

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. (2019). Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in neural information processing systems*, pages 8570–8581.

Lee, K., Lee, K., Lee, H., and Shin, J. (2018). A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing*

Systems, pages 7167–7177.

Li, Y., Ma, T., and Zhang, H. R. (2020). Learning over-parametrized two-layer neural networks beyond ntk. In *Conference on Learning Theory*, pages 2613–2682. PMLR.

Liang, S., Li, Y., and Srikant, R. (2017). Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*.

Liu, E. Z., Haghgoo, B., Chen, A. S., Raghunathan, A., Koh, P. W., Sagawa, S., Liang, P., and Finn, C. (2021). Just train twice: Improving group robustness without training group information. *ICML*.

Liu, Z., Luo, P., Wang, X., and Tang, X. (2015a). Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Liu, Z., Luo, P., Wang, X., and Tang, X. (2015b). Deep learning face attributes in the wild. *ICCV*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ma, C., Wang, K., Chi, Y., and Chen, Y. (2018). Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval and matrix completion. In *International Conference on Machine Learning*, pages 3345–3354. PMLR.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Mandt, S., Hoffman, M. D., and Blei, D. M. (2017). Stochastic gradient descent as approximate bayesian inference. *arXiv preprint arXiv:1704.04289*.

Marchenko, V. A. and Pastur, L. A. (1967). Distribution of eigenvalues for some sets of random matrices. *Matematicheskii Sbornik*, 114(4):507–536.

McCoy, R. T., Pavlick, E., and Linzen, T. (2019). Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Mei, S. and Montanari, A. (2019a). The generalization error of random features regression: precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*.

Mei, S. and Montanari, A. (2019b). The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*.

Metz, C. and Satariano, A. (2020). An algorithm that grants freedom, or takes it away. *The New York Times*, 6.

Mézard, M., Parisi, G., and Virasoro, M. (1987). *Spin glass theory and beyond: an introduction to the Replica Method and its applications*, volume 9. World Scientific Publishing Company.

Nagarajan, V., Andreassen, A., and Neyshabur, B. (2020). Understanding the failure modes of out-of-distribution generalization. *arXiv preprint arXiv:2010.15775*.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2019a). Deep double descent: where bigger models and more data hurt. *ICLR 2020, arXiv preprint arXiv:1912.02292*.

Nakkiran, P., Kaplun, G., Kalimeris, D., Yang, T., Edelman, B. L., Zhang, F., and Barak, B. (2019b). Sgd on neural networks learns functions of increasing complexity. *arXiv preprint*

arXiv:1905.11604.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. (2020). Learning from failure: Training debiased classifier from biased classifier. *arXiv preprint arXiv:2007.02561*.

Nar, K., Ocal, O., Sastry, S. S., and Ramchandran, K. (2019). Cross-entropy loss and low-rank features have responsibility for adversarial examples. *arXiv preprint arXiv:1901.08360*.

Nar, K. and Sastry, S. S. (2019). Persistency of excitation for robustness of neural networks. *arXiv preprint arXiv:1911.01043*.

Neal, B., Mittal, S., Baratin, A., Tantia, V., Scicluna, M., Lacoste-Julien, S., and Mitliagkas, I. (2018). A modern take on the bias-variance tradeoff in neural networks. *arXiv preprint arXiv:1810.08591*.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017a). Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. (2018). Towards understanding the role of over-parametrization in generalization of neural networks.

Neyshabur, B., Tomioka, R., Salakhutdinov, R., and Srebro, N. (2017b). Geometry of optimization and implicit regularization in deep learning. *arXiv preprint arXiv:1705.03071*.

Neyshabur, B., Tomioka, R., and Srebro, N. (2014). In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*.

Niven, T. and Kao, H.-Y. (2019). Probing neural network comprehension of natural language arguments. *arXiv preprint arXiv:1907.07355*.

Oakden-Rayner, L., Dunnmon, J., Carneiro, G., and Ré, C. (2020). Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, pages 151–159.

Opper, M. (1995). Statistical mechanics of learning: Generalization. *The handbook of brain theory and neural networks*, pages 922–925.

Opper, M. and Kinzel, W. (1996). Statistical mechanics of generalization. In *Models of neural networks III*, pages 151–209. Springer.

Oymak, S., Fabian, Z., Li, M., and Soltanolkotabi, M. (2019). Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*.

Parascandolo, G., Neitz, A., Orvieto, A., Gresele, L., and Schölkopf, B. (2020). Learning explanations that are hard to vary. *arXiv preprint arXiv:2009.00329*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

Pezeshki, M., Kaba, S.-O., Bengio, Y., Courville, A., Precup, D., and Lajoie, G. (2020). Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*.

Pezeshki, M., Mitra, A., Bengio, Y., and Lajoie, G. (2021). Multi-scale feature learning dynamics: Insights for double descent. *arXiv preprint arXiv:2112.03215*.

Pfungst, O. (1911). *Clever Hans:(the horse of Mr. Von Osten.) a contribution to experimental animal and human psychology*. Holt, Rinehart and Winston.

Poggio, T., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., Hidary, J., and Mhaskar, H. (2017). Theory of deep learning iii: explaining the non-overfitting puzzle. *arXiv preprint arXiv:1801.00173*.

Pohjonen, J., Stürenberg, C., Rannikko, A., Mirtti, T., and Pitkänen, E. (2021). Spectral decoupling allows training transferable neural networks in medical imaging. *arXiv preprint arXiv:2103.17171*.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. (2021). Grokking: Generalization beyond overfitting on small algorithmic datasets. In *ICLR MATH-AI Workshop*.

Raghu, M., Gilmer, J., Yosinski, J., and Sohl-Dickstein, J. (2017). Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR.

Rahmattalabi, A., Vayanos, P., Fulginiti, A., Rice, E., Wilder, B., Yadav, A., and Tambe, M. (2020). Exploring algorithmic fairness in robust graph covering problems. *arXiv preprint arXiv:2006.06865*.

Rauber, J., Brendel, W., and Bethge, M. (2017). Foolbox: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Roberts, M. (2021). Machine learning for covid-19 diagnosis: Promising, but still too flawed.

Roberts, M., Driggs, D., Thorpe, M., Gilbey, J., Yeung, M., Ursprung, S., Aviles-Rivero, A. I., Etmann, C., McCague, C., Beer, L., et al. (2021). Common pitfalls and recommendations for using machine learning to detect and prognosticate for covid-19 using chest radiographs and ct scans. *Nature Machine Intelligence*, 3(3):199–217.

Ronen, B., Jacobs, D., Kasten, Y., and Kritchman, S. (2019). The convergence rate of neural networks for learned functions of different frequencies. In *Advances in Neural Information Processing Systems*, pages 4761–4771.

Rosenfeld, A., Zemel, R., and Tsotsos, J. K. (2018). The elephant in the room. *arXiv preprint arXiv:1808.03305*.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. (2019). Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*.

Sagawa, S., Raghunathan, A., Koh, P. W., and Liang, P. (2020). An investigation of why overparameterization exacerbates spurious correlations. *arXiv preprint arXiv:2005.04345*.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2013a). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. cite arxiv:1312.6120.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2013b). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2019). A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546.

Sejnowski, T. J. (2018). *The deep learning revolution*. MIT press.

Seung, H. S., Sompolinsky, H., and Tishby, N. (1992). Statistical mechanics of learning from examples. *Physical review A*, 45(8):6056.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. (2020). The pitfalls of simplicity bias in neural networks. *arXiv preprint arXiv:2006.07710*.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shrestha, R., Kafle, K., and Kanan, C. (2022). An investigation of critical issues in bias mitigation techniques. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1943–1954.

Słowik, A. and Bottou, L. (2021). Algorithmic bias and data bias: Understanding the relation between distributionally robust optimization and data curation. *arXiv preprint arXiv:2106.09467*.

Sohoni, N. S., Dunnmon, J. A., Angus, G., Gu, A., and Ré, C. (2020). No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *arXiv preprint arXiv:2011.12945*.

Solla, S. A. (1995). A bayesian approach to learning in neural networks. *International Journal of Neural Systems*, 6:161–170.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. (2018a). The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. (2018b). The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57.

Spigler, S., Geiger, M., d'Ascoli, S., Sagun, L., Biroli, G., and Wyart, M. (2019). A jamming transition from under-to over-parametrization affects generalization in deep learning. *Journal of Physics A: Mathematical and Theoretical*, 52(47):474001.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Stephenson, C. and Lee, T. (2021). When and how epochwise double descent happens. *arXiv preprint arXiv:2108.12006*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Valle-Pérez, G., Camargo, C. Q., and Louis, A. A. (2018). Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*.

Vapnik, V. (1995). *The nature of statistical learning theory*. Springer science & business media.

Vapnik, V. N. (1998). *The nature of statistical learning theory*. Wiley, New York, 1st edition.

Vempala, S. and Wilmes, J. (2019). Gradient descent for one-hidden-layer neural networks: Polynomial convergence and sq lower bounds. In *Conference on Learning Theory*, pages 3115–3117.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset. Technical report.

Wang, H., He, Z., Lipton, Z. C., and Xing, E. P. (2019). Learning robust representations by projecting superficial statistics out. *arXiv preprint arXiv:1903.06256*.

Wang, S., Yu, X., and Perdikaris, P. (2021). When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, page 110768.

Watkin, T. L., Rau, A., and Biehl, M. (1993). The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65(2):499.

Williams, A., Nangia, N., and Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. *ACL*.

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390.

Wu, J., Hu, W., Xiong, H., Huan, J., Braverman, V., and Zhu, Z. (2020). On the noisy gradient descent that generalizes as sgd. In *International Conference on Machine Learning*, pages 10367–10376. PMLR.

Xu, X. and Frank, E. (2004). Logistic regression and boosting for labeled bags of instances. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 272–281. Springer.

Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y., and Ma, Z. (2019a). Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*.

Xu, Z.-Q. J., Zhang, Y., and Xiao, Y. (2019b). Training behavior of deep neural network in frequency domain. In *International Conference on Neural Information Processing*, pages 264–274. Springer.

Yang, G. and Salman, H. (2019). A fine-grained spectral perspective on neural networks. *arXiv preprint arXiv:1907.10599*.

Yang, Z., Yu, Y., You, C., Steinhardt, J., and Ma, Y. (2020). Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pages 10767–10777. PMLR.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Zech, J. R., Badgeley, M. A., Liu, M., Costa, A. B., Titano, J. J., and Oermann, E. K. (2018).

Confounding variables can degrade generalization performance of radiological deep learning models. *arXiv preprint arXiv:1807.00431*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

Zhang, X., Wu, D., Xiong, H., and Dai, B. (2021). Optimization variance: Exploring generalization properties of dnns. *arXiv preprint arXiv:2106.01714*.

Zhang, X., Xiong, H., and Wu, D. (2020). Rethink the connections among generalization, memorization and the spectral bias of dnns. *arXiv preprint arXiv:2004.13954*.

Zhao, J., Wang, T., Yatskar, M., Ordonez, V., and Chang, K.-W. (2017). Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *arXiv preprint arXiv:1707.09457*.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. (2020). Gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 109(3):467–492.

Supplementary Material For the First Article

.1. Further Related Work and Discussion

If we consider plots where the generalization error on the y-axis is plotted against other quantities on the x-axis, we find earlier works that have identified double descent behavior for quantities such as the number of parameters, the dimensionality of the data, the number of training samples, or the training time on the x-axis. In this paper, we studied epoch-wise double descent, *i.e.* we plot the training time t, or the number of training epochs, on the x-axis. Literature displaying double descent phenomena in generalization behavior w.r.t. other quantities do so in the limit of $t \to \infty$.

From a random matrix theory perspective, Le Cun et al. (1991); Hastie et al. (2019); Advani and Saxe (2017b), and Belkin et al. (2020) are among works which have analytically studied the spectral density of the Hessian matrix. According to their analyses, at intermediate levels of complexity, the presence of small but non-zero eigenvalues in the Hessian matrix results in high generalization error as the inverse of the Hessian is calculated for the pseudo-inverse solution.

Neyshabur et al. (2014) demonstrated that over-parameterized networks does not necessarily overfit thus suggesting the need of a new form of measure of model complexity other than network size. Subsequently, Neyshabur et al. (2018) suggest a novel complexity measure based on unit-wise capacities which correlates better with the behavior of test error with increasing network size. Chizat and Bach (2020) study the global convergence and superior generalization behavior of infinitely wide two-layer neural networks with logistic loss. Goldt et al. (2020) make use of the Gaussian Equivalence Theorem to study the generalization performance of two-layer neural networks and kernel models trained on data drawn from pre-trained generative models. Bai and Lee (2020) investigated the gap between the empirical performance of over-parameterized networks and their NTK counterparts, first proposed by Jacot et al. (2018).

From the perspective of bias/variance trade-off, Geman et al. (1992), and more recently, Neal et al. (2018) empirically observe that while bias is monotonically decreasing, variance could be decreasing too or unimodal as the number of parameters increases, thus manifesting a double descent generalization curve. Hastie et al. (2019) analytically study the variance. More recently, Yang et al. (2020) provides a new bias/variance decomposition of bias exhibiting double descent in which the variance follows a bell-shaped curve. However, the decrease in variance as the model size

increases remains unexplained. For high dimensional regression with random features, d'Ascoli et al. (2020) provides an asymptotic expression for the bias/variance decomposition and identifies three sources of variance with non-monotonous behavior as the model size or dataset size varies. d'Ascoli et al. (2020) also employs the analysis of random feature models and identifies two forms of overfitting which leads to the so-called sample-wise triple descent. More recently, Chen et al. (2020a) show that as a result of the interaction between the data and the model, one may design generalization curves with multiple descents.

From a statistical physics perspective, Opper (1995); Bös et al. (1993); Bös (1998); Opper and Kinzel (1996) are among the first studies which theoretically observe sample-wise double-descent in a ridge regression setup where the solution is obtained by the pseudo-inverse method. Most of these studies employ the "Gardner analysis" (Gardner, 1988; Gardner and Derrida, 1988, 1989) for models where the number of parameters and the dimensionality of data are coupled and hence the observed form of double descent is different from that observed in deep neural networks. A beautiful extended review of this line of work is provided in Engel and Van den Broeck (2001). Among recent works, Gerace et al. (2020) also apply the Gardner analysis but to a novel generalized data generating process called the hidden manifold model and derive the model-wise double-descent equations analytically.

Finally, recall that towards providing an explanation for the epoch-wise double descent, we argue that *the epoch-wise double descent can be attributed to different features being learned at different time-scales*, resulting in a non-monotonous generalization curve. In relation to the aspect of different feature learning scales, Rahaman et al. (2019) had observed that DNNs have a tendency towards learning simple target functions first that can allow for good generalization behavior of various data samples. Pezeshki et al. (2020) also identify and provide explanation for a feature learning imbalance exhibited by over-parameterized networks trained via gradient descent on cross-entropy loss, with the networks learning only a subset of the full feature spectrum over training. More recently, Zhang et al. (2020), show that certain DNNs models prioritize learning high-frequency components first followed by the learning of slow but informative features, leading to the second descent of the test error as observed in epoch-wise double descent.

On the difference between model-wise and epoch-wise double descent curves. In accordance with its name, model-wise double descent (in the test error) occurs due to an increase in model-size (number of its parameters), i.e., as the model transitions from an under-parameterized to an over-parameterized regime. A variety of works have tried to understand this phenomenon from the lens of implicit regularization (Neyshabur et al., 2014) or defining novel complexity measures (Neyshabur et al., 2017a). On the other hand, epoch-wise double descent (in the test error) as treated in our work, is observed to occur for both over-parameterized (Nakkiran et al., 2019a) and under-parameterized (Heckel and Yilmaz, 2020) setups. As found in our work along with the latter reference, this phenomenon seems to be a result of different feature learning speeds rather than the extent of model parameterization. The overlap of the test-error contributions from the different

weights with varying scales of learning henceforth leads to a non-monotonous evolution of the model test error as exemplified by epoch-wise double descent.

We also note that the peak in model-wise double descent is associated with the model's capacity to perfectly interpolate the data, we do not think an analogous notion exists for the case of epoch-wise double descent. Our understanding of the peak in the latter is that it corresponds to a training time configuration whereby a subclass of features are already learnt (due to a larger associated signal-to-noise-ratio) and are being overfitted upon to fit the target. As training proceeds further, the remaining set of features are eventually learnt thus allowing for a lowering of the test error.

On the link to complex networks. Generally, exact study of complex neural networks is often intractable. A common practice is to study a simpler system that conserves key attributes and then validate the findings on the original complex system. In this work, we build on the same established practices: we propose a simple linear model with two key advantages, a) it can be solved analytically, b) exhibits double descent, the property of interest. Subsequently, our experiments support the extension of our findings and intuitions to complex neural networks.

.2. Technical Proofs

.2.1. The generalization error as a function of R and Q (Eq. 4.2.6)

Recall that the teacher is the data generator and is defined as,

$$y := y^* + \epsilon, \qquad y^* := \boldsymbol{z}^T \boldsymbol{W}, \qquad z_i \sim \mathcal{N}(0, \frac{1}{\sqrt{d}}), \tag{.2.1}$$

where $\boldsymbol{z} \in \mathbb{R}^d$ is the teacher's input and $y^*, y \in \mathbb{R}$ are the teacher's noiseless and noisy outputs, respectively. $\boldsymbol{W} \in \mathbb{R}^d$ represents the (fixed) weights of the teacher and $\epsilon \in \mathbb{R}$ is the label noise.

While the student network is defined as,

$$\hat{y} := \boldsymbol{x}^T \hat{\boldsymbol{W}}, \qquad s.t. \qquad \boldsymbol{x} := \mathrm{F}^T \boldsymbol{z}, \tag{.2.2}$$

where the matrix $\mathrm{F} \in \mathbb{R}^{d \times d}$ is a predefined and fixed modulation matrix regulating the student's access to the true input $\boldsymbol{z}$.

The average generalization error of the student, determined by averaging the student's error over all possible input configurations and label noise realizations is given by,

$$\mathcal{L}_G := \frac{1}{2} \mathbb{E}_{\boldsymbol{z},\epsilon} \Big[(y^* - \hat{y} + \epsilon)^2 \Big], \tag{.2.3}$$

in which the variables $(y^*, \hat{y})$ form a bi-variate Gaussian distribution with zero mean and a covariance of,

$$\Sigma = \begin{bmatrix} < y^*, y^* >_{\boldsymbol{z}} & < y^*, \hat{y} >_{\boldsymbol{z}} \\ < y^*, \hat{y} >_{\boldsymbol{z}} & < \hat{y}, \hat{y} >_{\boldsymbol{z}} \end{bmatrix} = \begin{bmatrix} 1 & R \\ R & Q \end{bmatrix}, \tag{.2.4}$$

Here,

$$R := \mathbb{E}_{\boldsymbol{z}}[y^* \hat{y}] = \mathbb{E}_{\boldsymbol{z}}[\boldsymbol{W}^T \boldsymbol{z}\boldsymbol{z}^T \mathrm{F} \hat{\boldsymbol{W}}] = \frac{1}{d} \boldsymbol{W}^T \mathrm{F} \hat{\boldsymbol{W}}, \quad \text{and,} \tag{.2.5}$$

$$Q := \mathbb{E}_{\boldsymbol{z}}[\hat{y}^T \hat{y}] = \mathbb{E}_{\boldsymbol{z}}[\hat{\boldsymbol{W}}^T \mathrm{F}^T \boldsymbol{z}\boldsymbol{z}^T \mathrm{F} \hat{\boldsymbol{W}}] = \frac{1}{d} \hat{\boldsymbol{W}}^T \mathrm{F}^T \mathrm{F} \hat{\boldsymbol{W}}. \tag{.2.6}$$

Utilizing this, Eq. .2.3 can be expressed as,

$$\mathcal{L}_G := \frac{1}{2} \mathbb{E}_{\boldsymbol{z}} \left[(y^* - \hat{y} + \epsilon)^2 \right], \tag{.2.7}$$

$$= \frac{1}{2} \mathbb{E}_{\tilde{y}^*, \tilde{\hat{y}}} \left[(\tilde{y}^* - (R\tilde{y}^* + \sqrt{Q - R^2} \tilde{\hat{y}}) + \epsilon)^2 \right], \tag{.2.8}$$

$$= \frac{1}{2} (1 + \epsilon^2 + Q - 2R). \tag{.2.9}$$

Additionally, we note that expectation w.r.t. a Gaussian variable x is defined as,

$$\mathbb{E}_x[f(x)] := \int_{-\infty}^{+\infty} \frac{dx}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) f(x). \tag{.2.10}$$

.2.2. The general case exact dynamics (Eqs. 4.2.9-4.2.10)

Recall that to train our student network, we use gradient descent (GD) on the regularized mean-squared loss, evaluated on the n training examples as,

$$\mathcal{L}_T := \frac{1}{2n} \sum_{\mu=1}^{n} (y^\mu - \hat{y}^\mu)^2 + \frac{\lambda}{2} ||\hat{\boldsymbol{W}}||_2^2, \tag{.2.11}$$

where $\lambda \in [0, \infty)$ is the regularization coefficient.

The minimum of the loss function, denoted by $\hat{\boldsymbol{W}}_{\text{gd}}$, is achieved at,

$$\nabla_{\hat{\boldsymbol{W}}} \mathcal{L}_T = 0 \Rightarrow \nabla_{\hat{\boldsymbol{W}}} \left[\frac{1}{2} ||\boldsymbol{y} - \mathrm{X} \hat{\boldsymbol{W}}||_2^2 + \frac{\lambda}{2} ||\hat{\boldsymbol{W}}||_2^2 \right] = 0 \tag{.2.12}$$

$$\Rightarrow -\mathrm{X}^T (\boldsymbol{y} - \mathrm{X} \hat{\boldsymbol{W}}_{\text{gd}}) + \lambda \hat{\boldsymbol{W}}_{\text{gd}} = 0 \tag{.2.13}$$

$$\Rightarrow \hat{\boldsymbol{W}}_{\text{gd}} := (\mathrm{X}^T \mathrm{X} + \lambda \mathrm{I})^{-1} \mathrm{X}^T y. \tag{.2.14}$$

Additionally, the exact dynamics under gradient-descent, correspond to,

$$\begin{aligned}
\hat{\boldsymbol{W}}_t &= \hat{\boldsymbol{W}}_{t-1} - \eta \nabla_{\hat{\boldsymbol{W}}_{t-1}} \mathcal{L}_T, \\
&= \hat{\boldsymbol{W}}_{t-1} - \eta \Big[-\mathrm{X}^T(\boldsymbol{y} - \mathrm{X}\hat{\boldsymbol{W}}_{t-1}) + \lambda \hat{\boldsymbol{W}}_{t-1} \Big] \\
&= (1-\eta\lambda)\hat{\boldsymbol{W}}_{t-1} - \eta \mathrm{X}^T\mathrm{X}\hat{\boldsymbol{W}}_{t-1} + \eta \mathrm{X}^T\boldsymbol{y}, \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]\hat{\boldsymbol{W}}_{t-1} + \eta \mathrm{X}^T\boldsymbol{y}, \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]\hat{\boldsymbol{W}}_{t-1} + \eta (\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I})(\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I})^{-1}\mathrm{X}^T\boldsymbol{y}, \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]\hat{\boldsymbol{W}}_{t-1} + \eta (\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I})\hat{\boldsymbol{W}}_{\text{gd}}, \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]\hat{\boldsymbol{W}}_{t-1} + (\eta \mathrm{X}^T\mathrm{X} + \eta\lambda\mathrm{I})\hat{\boldsymbol{W}}_{\text{gd}}, \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]\hat{\boldsymbol{W}}_{t-1} + (\eta \mathrm{X}^T\mathrm{X} + (\eta\lambda - 1)\mathrm{I})\hat{\boldsymbol{W}}_{\text{gd}} + \hat{\boldsymbol{W}}_{\text{gd}},
\end{aligned} \tag{.2.15}$$

which leads to,

$$\begin{aligned}
\hat{\boldsymbol{W}}_t - \hat{\boldsymbol{W}}_{\text{gd}} &= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}](\hat{\boldsymbol{W}}_{t-1} - \hat{\boldsymbol{W}}_{\text{gd}}), \\
&= [(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X}]^t(\hat{\boldsymbol{W}}_0 - \hat{\boldsymbol{W}}_{\text{gd}}).
\end{aligned} \tag{.2.16}$$

Assuming $\hat{\boldsymbol{W}}_0 = 0$, we arrive at the following closed-form equation,

$$\hat{\boldsymbol{W}}_t = \left(\mathrm{I} - \Big[(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X} \Big]^t \right) \hat{\boldsymbol{W}}_{\text{gd}}, \tag{.2.17}$$

where $\hat{\boldsymbol{W}}_{\text{gd}}$ is defined in Eq .2.14.

Now back to definition of R in Eq. .2.5 and by substitution of Eq. .2.17, we have,

$$\begin{aligned}
R(t) :&= \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\hat{\boldsymbol{W}}_t, \\
&= \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\left(\mathrm{I} - \Big[(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X} \Big]^t \right) \hat{\boldsymbol{W}}_{\text{gd}}, \\
&= \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\left(\mathrm{I} - \Big[(1-\eta\lambda)\mathrm{I} - \eta \mathrm{X}^T\mathrm{X} \Big]^t \right) (\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I})^{-1}\mathrm{X}^T\boldsymbol{y}, \\
&= \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\mathrm{V}\left(\mathrm{I} - [(1-\eta\lambda)\mathrm{I} - \eta\Lambda]^t \right) (\Lambda + \lambda\mathrm{I})^{-1}\mathrm{V}^T\mathrm{X}^T\boldsymbol{y}, \quad (\mathrm{X}^T\mathrm{X} = \mathrm{V}\Lambda\mathrm{V}^T) \\
&= \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\mathrm{V}\left(\mathrm{I} - [(1-\eta\lambda)\mathrm{I} - \eta\Lambda]^t \right) (\Lambda + \lambda\mathrm{I})^{-1}(\Lambda\mathrm{V}^T\mathrm{F}^{-1}\boldsymbol{W} + \Lambda^{\frac{1}{2}}\boldsymbol{\epsilon}), \\
&= \boxed{\frac{1}{d}\mathbf{Tr}\left[\left(\mathrm{I} - [(1-\eta\lambda)\mathrm{I} - \eta\Lambda]^t \right) \frac{\Lambda}{\Lambda + \lambda\mathrm{I}} \right]}.
\end{aligned} \tag{.2.18}$$

Similarly for Q, let $D := \Big(\mathrm{I} - [(1-\eta\lambda)\mathrm{I} - \eta\Lambda]^t\Big)$, then we have,

$$
\begin{aligned}
Q(t) :&= \frac{1}{d}\hat{\boldsymbol{W}}^T\mathrm{F}^T\mathrm{F}\hat{\boldsymbol{W}}, \\
&= \frac{1}{d}\hat{\boldsymbol{W}}_{\text{gd}}^T\Big(\mathrm{I} - \Big[(1-\eta\lambda)\mathrm{I} - \eta\mathrm{X}^T\mathrm{X}\Big]^t\Big)\mathrm{F}^T\mathrm{F}\Big(\mathrm{I} - \Big[(1-\eta\lambda)\mathrm{I} - \eta\mathrm{X}^T\mathrm{X}\Big]^t\Big)\hat{\boldsymbol{W}}_{\text{gd}}, \\
&= \frac{1}{d}\hat{\boldsymbol{W}}_{\text{gd}}^T\mathrm{V}\mathrm{D}\mathrm{V}^T\mathrm{F}^T\mathrm{F}\mathrm{V}\mathrm{D}\mathrm{V}^T\hat{\boldsymbol{W}}_{\text{gd}}, \\
&= \frac{1}{d}\hat{\boldsymbol{W}}_{\text{gd}}^T\mathrm{V}\mathrm{D}\tilde{\mathrm{F}}^T\tilde{\mathrm{F}}\mathrm{D}\mathrm{V}^T\hat{\boldsymbol{W}}_{\text{gd}}, \qquad (\tilde{\mathrm{F}} := \mathrm{F}\mathrm{V}, \mathrm{X} = \mathrm{U}\Lambda^{1/2}\mathrm{V}^T, \tilde{\boldsymbol{\epsilon}} := \mathrm{U}^T\boldsymbol{\epsilon}) \\
&= \frac{1}{d}(\boldsymbol{W}^T\mathrm{F}^{-1^T}\mathrm{V} + \Lambda^{-1/2}\tilde{\epsilon})\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\mathrm{D}\tilde{\mathrm{F}}^T\tilde{\mathrm{F}}\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}(\mathrm{V}^T\mathrm{F}^{-1}\boldsymbol{W} + \Lambda^{-1/2}\tilde{\boldsymbol{\epsilon}}), \qquad (.2.19) \\
&= \frac{1}{d}(\boldsymbol{W}^T\tilde{\mathrm{F}}^{-1^T} + \Lambda^{-1/2}\tilde{\boldsymbol{\epsilon}})\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\mathrm{D}\tilde{\mathrm{F}}^T\tilde{\mathrm{F}}\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}(\tilde{\mathrm{F}}^{-1}\boldsymbol{W} + \Lambda^{-1/2}\tilde{\boldsymbol{\epsilon}}), \\
&= \frac{1}{d}\boldsymbol{W}^T\tilde{\mathrm{F}}^{-1^T}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\mathrm{D}\tilde{\mathrm{F}}^T\tilde{\mathrm{F}}\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\tilde{\mathrm{F}}^{-1}\boldsymbol{W}, \\
&\quad + \frac{1}{d}\Lambda^{-1/2}\tilde{\boldsymbol{\epsilon}}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\mathrm{D}\tilde{\mathrm{F}}^T\tilde{\mathrm{F}}\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\Lambda^{-1/2}\tilde{\boldsymbol{\epsilon}}, \\
&= \boxed{\frac{1}{d}\mathbf{Tr}\left[\mathrm{A}^T\mathrm{A}\right] + \frac{\sigma_\epsilon^2}{d}\mathbf{Tr}\left[\mathrm{B}^T\mathrm{B}\right]}
\end{aligned}
$$

where,

$$
\mathrm{A} := \tilde{\mathrm{F}}\,\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\tilde{\mathrm{F}}^{-1} \quad \text{and,} \quad \mathrm{B} := \tilde{\mathrm{F}}\,\mathrm{D}\frac{\Lambda}{\Lambda+\lambda\mathrm{I}}\Lambda^{-\frac{1}{2}}. \qquad (.2.20)
$$

.2.3. Special case of approximate dynamics (Eqs. 4.2.14 and 4.2.15)

Recall that the teacher and student are defined as,

$$
y := y^* + \epsilon, \qquad y^* := \boldsymbol{z}^T\boldsymbol{W}, \qquad \hat{y} := \boldsymbol{x}^T\hat{\boldsymbol{W}}, \qquad \boldsymbol{x} := \mathrm{F}^T\boldsymbol{z}, \qquad (.2.21)
$$

where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is the label noise, F is the modulation matrix, and $||\boldsymbol{z}||_2^2 = ||\boldsymbol{W}||_2^2 = 1$.

The training and generalization losses are defined as,

$$
\mathcal{L}_T := \frac{1}{2n}\sum(\hat{y}-y)^2 + \frac{\lambda}{2}||\hat{\boldsymbol{W}}||_2^2, \qquad \mathcal{L}_G := \frac{1}{2}\mathbb{E}_{\boldsymbol{z},\epsilon}[(\hat{y}-y)^2]. \qquad (.2.22)
$$

According to Eq. 4.2.6, the generalization loss can be written in terms of two scalar variables R and Q,

$$
\mathcal{L}_G = \frac{1}{2}(1 + \sigma_\epsilon^2 + Q - 2R), \quad \text{where,} \qquad (.2.23)
$$

$$
R := \mathbb{E}_{\boldsymbol{z}}[y^*\hat{y}] = \mathbb{E}_{\boldsymbol{z}}[\boldsymbol{W}^T\boldsymbol{z}\boldsymbol{z}^T\mathrm{F}\hat{\boldsymbol{W}}] = \frac{1}{d}\boldsymbol{W}^T\mathrm{F}\hat{\boldsymbol{W}}, \quad \text{and,} \qquad (.2.24)
$$

$$
Q := \mathbb{E}_{\boldsymbol{z}}[\hat{y}\hat{y}] = \mathbb{E}_{\boldsymbol{z}}[\hat{\boldsymbol{W}}^T\mathrm{F}^T\boldsymbol{z}\boldsymbol{z}^T\mathrm{F}\hat{\boldsymbol{W}}] = \frac{1}{d}\hat{\boldsymbol{W}}^T\mathrm{F}^T\mathrm{F}\hat{\boldsymbol{W}}. \qquad (.2.25)
$$

In the following, we next determine the most probable values of the above scalar entities, from statistical perspective.

Application of t steps of GD on $\mathcal{L}_T$ results in the following distribution for the student's weights:

$$P(\hat{\boldsymbol{W}}, t) = \frac{1}{Z_{\beta,t}} e^{-\beta \tilde{\mathcal{L}}_T(\hat{\boldsymbol{W}}, t)}, \tag{.2.26}$$

in which $\tilde{\mathcal{L}}_T(\hat{\boldsymbol{W}}, t)$ is a modified loss that dictates the distribution of student weights $\hat{\boldsymbol{W}}$ upon t^{th} iterations of GD on the original loss $\mathcal{L}_T(\hat{\boldsymbol{W}})$, while β corresponds to an (inverse) temperature parameter of our student weight distribution.

In Eq. .2.26, $Z_{\beta,t}$ is the partition function which is defined as,

$$Z_{\beta,t} = \frac{\int_{-\infty}^{\infty} \prod_{i=1}^{d} \mathrm{d}\Big(\hat{\boldsymbol{W}}_i\Big) \delta\Big(\frac{1}{d} \hat{\boldsymbol{W}}_i^T \mathrm{F}^T \mathrm{F} \hat{\boldsymbol{W}}_i - Q_0\Big) e^{-\beta \tilde{\mathcal{L}}_T(\hat{\boldsymbol{W}}, t)}}{\int_{-\infty}^{\infty} \prod_{i=1}^{d} \mathrm{d}\Big(\hat{\boldsymbol{W}}_i\Big) \delta\Big(\frac{1}{d} \hat{\boldsymbol{W}}_i^T \mathrm{F}^T \mathrm{F} \hat{\boldsymbol{W}}_i - Q_0\Big)}, \tag{.2.27}$$

in which, Q_0 can be perceived to be a target norm the student weights $\hat{\boldsymbol{W}}$ are being constrained to and d is the dimensionality of the data.

We are now interested in finding R and Q of the typical (most probable) students. Therefore, it suffices to find the students that dominate the partition function (or more precisely the free-energy). The free-energy is defined as,

$$f := -\frac{1}{\beta d} \mathbb{E}_{\boldsymbol{W}, \boldsymbol{z}} \Big[\ln Z_{\beta,t} \Big], \tag{.2.28}$$

where $\boldsymbol{W}$ and $\boldsymbol{z}$ are the teacher's weight and input, respectively.

Due to the logarithm inside the expectation, analytical computation of Eq. .2.28 is intractable. However, the replica method (Mézard et al., 1987) allows us to tackle this through the following identity,

$$\mathbb{E}_{\boldsymbol{W}, \boldsymbol{z}}[\ln Z_{\beta,t}] = \lim_{r \to 0} \frac{\mathbb{E}_{\boldsymbol{W}, \boldsymbol{z}}[Z_{\beta,t}^r] - 1}{r}. \tag{.2.29}$$

Case 1: F = I. As a first step, we first study a case where $\mathrm{F} = \mathrm{I}$. In that case, as derived in Bös (1998), Eq. .2.28 can be simplified to,

$$-\beta f = \frac{1}{2} \frac{Q - R^2}{Q_0 - Q} + \frac{1}{2} \ln(Q_0 - Q) - \frac{n}{2d} \ln[1 + \beta(Q_0 - Q)] - \frac{n\beta}{2d} \frac{G - 2HR + Q}{1 + \beta(Q_0 - Q)}, \tag{.2.30}$$

in which the scalar variables G and H are defined as,

$$H := \mathbb{E}_{y^*, \epsilon}[y^* y] = \mathbb{E}_{y^*}[y^*(y^* + \epsilon)] = 1, \tag{.2.31}$$

$$G := \mathbb{E}_{y^*, \epsilon}[yy] = \mathbb{E}_{y^*}[(y^* + \epsilon)(y^* + \epsilon)] = 1 + \sigma_\epsilon^2. \tag{.2.32}$$

At this point, in order to find the most probable students, one can extremize the free-energy $f(R, Q, Q_0)$ in Eq. .2.30. The solution to this extremisation is derived in Bös et al. (1993) and

reads,

$$\nabla_R f = 0 \qquad \Rightarrow \qquad R = \frac{n}{d}\frac{1}{a}, \tag{.2.33}$$

$$\nabla_Q f = 0 \qquad \Rightarrow \qquad Q = \frac{n}{d}\frac{1}{a^2 - n/d}\left(G - \frac{n}{d}\frac{2-a}{a}\right), \tag{.2.34}$$

$$\nabla_{Q_0} f = 0 \qquad \Rightarrow \qquad a = 1 + \frac{2\tilde{\lambda}}{1 - n/d - \tilde{\lambda} + \sqrt{(1 - n/d - \tilde{\lambda})^2 + 4\tilde{\lambda}}}, \tag{.2.35}$$

in which,

$$a := 1 + \frac{1}{\beta(Q_0 - Q)}, \qquad \text{and,} \qquad \tilde{\lambda} := \lambda + \frac{1}{\eta t}. \tag{.2.36}$$

Case 2: F follows Assumption 4.2.1. The modulation matrix, F, under a SVD, $\mathrm{F} := \mathrm{U}\Sigma\mathrm{V}^T$ has two sets of singular values such that the first p singular values are equal to σ_1 and the remaining $d - p$ singular values are equal to σ_2. We let the condition number of F to be denoted by $\kappa := \frac{\sigma_1}{\sigma_2} > 1$.

Without loss of generality, we hereby assume that $\mathrm{U} = \mathrm{V} = \mathrm{I}$. Consequently, the (noiseless) teacher and the student can be written as the composition of two sub-models as following,

$$y^* = y_1^* + y_2^* = \boldsymbol{z}_1^T \boldsymbol{W}_1 + \boldsymbol{z}_2^T \boldsymbol{W}_2, \qquad \text{(teacher decomposition)} \tag{.2.37}$$

$$\hat{y} = \hat{y}_1 + \hat{y}_2 = \sigma_1 \boldsymbol{z}_1^T \hat{\boldsymbol{W}}_1 + \sigma_2 \boldsymbol{z}_2^T \hat{\boldsymbol{W}}_2, \qquad \text{(student decomposition)} \tag{.2.38}$$

in which $\boldsymbol{z}_1 \in \mathbb{R}^p$ and $\boldsymbol{z}_2 \in \mathbb{R}^{d-p}$.

Let $\hat{y}_i$ denote the output of the i^{th} component of the student. Also let y_i^* and y_i denote the noiseless and noisy targets, respectively. Therefore, for the student components $i \in 1, 2$, we have,

$$\begin{aligned} \hat{y}_1 &= \sigma_1 z_1^T \hat{\boldsymbol{W}}_1, \\ y_1^* &= z_1^T W_1, \\ y_1 &= y_1^* + \underbrace{z_2^T W_2 - \sigma_2 z_2^T \hat{\boldsymbol{W}}_2}_{y_2^* - \hat{y}_2 = \epsilon_2(t)} + \epsilon, \end{aligned} \qquad \left| \qquad \begin{aligned} \hat{y}_2 &= \sigma_2 z_2^T \hat{\boldsymbol{W}}_2, \\ y_2^* &= z_2^T W_2, \\ y_2 &= y_2^* + \underbrace{z_1^T W_1 - \sigma_1 z_1^T \hat{\boldsymbol{W}}_1}_{y_1^* - \hat{y}_1 = \epsilon_1(t)} + \epsilon, \end{aligned} \right.$$

in which ϵ is the *explicit noise*, added to the teacher's output while $\epsilon_j(t)$ is an *implicit variable noise* which decreases as the component $j \neq i$ learns to match $\hat{y}_j$ and y_j.

Accordingly, the variables H_i and G_i for each component i are re-defined as,

$$
\begin{aligned}
H_1 &= \mathbb{E}[y_1^* y_1] = \mathbb{E}_{y_1^*}[y_1^* y_1^*] = \frac{p}{d}, \\
G_1 &= \mathbb{E}[y_1 y_1], \\
&= \mathbb{E}[(y_1^* + y_2^* - \hat{y}_2)(y_1^* + y_2^* - \hat{y}_2)] + \sigma_\epsilon^2, \\
&= \mathbb{E}[y_1^* y_1^*] + \mathbb{E}[y_2^* y_2^*] + \mathbb{E}[\hat{y}_2 \hat{y}_2], \\
&\quad - 2\mathbb{E}[y_2^* \hat{y}_2] + \sigma_\epsilon^2, \\
&= \frac{p}{d} + \frac{d-p}{d} + Q_2 - 2R_2 + \sigma_\epsilon^2, \\
&= 1 + Q_2 - 2R_2 + \sigma_\epsilon^2,
\end{aligned}
\qquad
\begin{aligned}
H_2 &= \mathbb{E}[y_2^* y_2] = \mathbb{E}_{y_2^*}[y_2^* y_2^*] = \frac{d-p}{d}, \\
G_2 &= \mathbb{E}[y_2^T y_2], \\
&= \mathbb{E}[(y_2^* + y_1^* - \hat{y}_1)^T (y_2^* + y_1^* - \hat{y}_1)] + \sigma_\epsilon^2, \\
&= \mathbb{E}[y_2^* y_2^*] + \mathbb{E}[y_1^* y_1^*] + \mathbb{E}[\hat{y}_1 \hat{y}_1], \\
&\quad - 2\mathbb{E}[y_1^* \hat{y}_1] + \sigma_\epsilon^2, \\
&= \frac{d-p}{d} + \frac{p}{d} + Q_1 - 2R_1 + \sigma_\epsilon^2, \\
&= 1 + Q_1 - 2R_1 + \sigma_\epsilon^2,
\end{aligned}
$$

in which R_i and Q_i are defined as,

$$
R_i := \mathbb{E}_z[y_i^* \hat{y}_i] = \frac{1}{d} W_i^T \sigma_i \hat{\boldsymbol{W}}_i, \quad \text{and,} \quad Q_i := \mathbb{E}_z[\hat{y}_i \hat{y}_i] = \frac{1}{d} \hat{\boldsymbol{W}}_i^T \sigma_i^2 \hat{\boldsymbol{W}}_i,
$$

where σ_i denotes the singular values of the matrix F as defined in Assumption 4.2.1.

Rewriting Eqs. .2.33, .2.34, and .2.35 for each of the student's components, we arrive at,

$$
\begin{aligned}
R_1 &= \frac{n}{d}\frac{1}{a_1}, \\
Q_1 &= \frac{n}{p a_1^2 - n}\left(1 + Q_2 - 2R_2 + \sigma_\epsilon^2 - \frac{n}{d}\frac{2 - a_1}{a_1}\right), \\
a_1 &= 1 + \frac{2\tilde{\lambda}_1}{1 - \frac{n}{p} - \tilde{\lambda}_1 + \sqrt{(1 - \frac{n}{p} - \tilde{\lambda}_1)^2 + 4\tilde{\lambda}_1}}, \\
\tilde{\lambda}_1 &:= \frac{d}{p}\frac{1}{\sigma_1^2}(\lambda + \frac{1}{\eta t}),
\end{aligned}
\qquad
\begin{aligned}
R_2 &= \frac{n}{d}\frac{1}{a_2}, \\
Q_2 &= \frac{n}{(d-p) a_1^2 - n}\left(1 + Q_1 - 2R_1 + \sigma_\epsilon^2 - \frac{n}{d}\frac{2 - a_2}{a_2}\right), \\
a_2 &= 1 + \frac{2\tilde{\lambda}}{1 - \frac{n}{d-p} - \tilde{\lambda} + \sqrt{(1 - \frac{n}{d-p} - \tilde{\lambda})^2 + 4\tilde{\lambda}}}, \\
\tilde{\lambda}_2 &:= \frac{d}{d-p}\frac{1}{\sigma_2^2}(\lambda + \frac{1}{\eta t}),
\end{aligned}
$$

where Q_1 depends on Q_2 and vice versa. However, with simple calculations, we can arrive at the following standalone equation. Let,

$$
\alpha_1 = \frac{n}{p}, \alpha_2 = \frac{n}{d-p}, \tag{.2.39}
$$

and also let,

$$
b_i = \frac{\alpha_i}{a_i^2 - \alpha_i}, \quad c_i = 1 - 2R_i - \frac{n}{d}\frac{2 - a_i}{a_i} \qquad \text{for} \qquad i \in \{1, 2\}, \tag{.2.40}
$$

with which the closed-from scalar expression for $Q(t, \lambda)$ reads,

$$
Q(t, \lambda) = Q_1 + Q_2, \quad \text{where,} \quad Q_1 := \frac{b_1 b_2 c_2 + b_1 c_1}{1 - b_1 b_2}, \quad \text{and,} \quad Q_2 := \frac{b_1 b_2 c_1 + b_2 c_2}{1 - b_1 b_2}. \tag{.2.41}
$$

.2.4. Derivation of $\tilde{\mathcal{L}}(\hat{\boldsymbol{W}}, t)$ in Eq. 4.2.22.

The goal is to show,

$$\hat{\boldsymbol{W}}_t = \arg\min_{\hat{\boldsymbol{W}}} \tilde{\mathcal{L}}_T(\hat{\boldsymbol{W}}, t), \qquad \text{where,} \qquad \hat{\boldsymbol{W}}_t := \hat{\boldsymbol{W}}_{t-1} - \eta \nabla_{\hat{\boldsymbol{W}}_{t-1}} \mathcal{L}(\hat{\boldsymbol{W}}_{t-1}). \tag{.2.42}$$

For brevity of derivations, here we only consider the case where $\lambda = \sigma_\epsilon^2 = 0$. Recall the closed-form derivation of $\hat{\boldsymbol{W}}_t$ in Eq. 4.2.19,

$$\hat{\boldsymbol{W}}_t = \Big(I - \Big[I - \eta X^T X\Big]^t\Big)(X^T X)^{-1} X^T y, \tag{.2.43}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \Big[X\hat{\boldsymbol{W}} - X\Big(I - \Big[I - \eta X^T X\Big]^t\Big)(X^T X)^{-1} X^T y\Big]^2, \tag{.2.44}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T}\Big(I - \Big[I - \eta X^T X\Big]^t\Big) \underbrace{(X^T X)^{-1} X^T y}_{=W, \text{ assuming } \sigma_\epsilon^2 = 0}\Big]^2, \tag{.2.45}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} V\Big(I - \Big[I - \eta\Lambda\Big]^t\Big) V^T W\Big]^2, \qquad (X^T X = V\Lambda V^T) \tag{.2.46}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} V\Big(I - \exp\Big(t \log[I - \eta\Lambda]\Big)\Big) V^T W\Big]^2, \tag{.2.47}$$

$$\approx \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} V\Big(I - \exp\big(-\eta\Lambda t\big)\Big) V^T W\Big]^2, \qquad (\log(1+x) \approx x) \tag{.2.48}$$

$$\approx \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} V\Big(I - \exp\Big(-\log\Big(\frac{\Lambda}{1/\eta t} + I\Big)\Big)\Big) V^T W\Big]^2, \quad (\log(1+x) \approx x) \tag{.2.49}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} V\Big(I - \Big[\Lambda + \frac{1}{\eta t} I\Big]^{-1} \frac{1}{\eta t}\Big) V^T W\Big]^2, \tag{.2.50}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \frac{1}{2n} \sum \Big[\hat{y}^\mu - x^{\mu^T} (X^T X + \frac{1}{\eta t} I)^{-1} X^T X W\Big]^2, \tag{.2.51}$$

$$= \arg\min_{\hat{\boldsymbol{W}}} \underbrace{\frac{1}{2n} \sum \Big[\hat{y}^\mu - y^\mu\Big]^2 + \frac{1}{\eta t} ||\hat{\boldsymbol{W}}||_2^2}_{\tilde{\mathcal{L}}_T(\hat{\boldsymbol{W}}, t)}, \tag{.2.52}$$

which concludes the proof.

This proof have a core dependence on the findings of Ali et al. (2019, 2020). These works first formalize the connection between (continuous-time) GD or SGD-based training of an ordinary least squares (OLS) setup and that of ridge regression, providing bounds on the test error under these algorithms over training time t, in terms of a ridge setup with ridge parameter $\lambda = 1/t$. We utilize these results in the sense that by evaluating the generalization error $\mathcal{L}_G$ of our student-teacher setup

with explicit ridge regularization, we invoke the connection between the ridge coefficient λ and training time t as described in these works, to obtain the behavior of (ridgeless) $\mathcal{L}_G$ over training.

.2.5. Proof of Lemma 4.3.1

For a linear/linearized model, penalizing Q amounts to adding the following regularizer to the loss,

$$\mathcal{L}_T \leftarrow \mathcal{L}_T + \alpha ||\hat{y}||^2,$$

previously introduced in Pezeshki et al. (2020).

Proof: Recall that the variable Q is defined as,

$$Q := \frac{1}{d} \hat{W}^T \mathrm{F}^T \mathrm{F} \hat{W}.$$

Since Z is normally distributed with unit covariance, we can rewrite Q as,

$$Q := \frac{1}{d} \hat{W}^T \mathrm{F}^T Z^T Z \mathrm{F} \hat{W} = \frac{1}{d} \hat{W}^T X^T X \hat{W}.$$

We note that for a linear/linearized model of form $\hat{\boldsymbol{y}} := X^T \hat{W}$, the following identity holds,

$$||\hat{\boldsymbol{y}}||^2 = \hat{\boldsymbol{y}}^T \hat{\boldsymbol{y}} = \hat{W}^T X^T X \hat{W} = dQ.$$

.2.6. Replica Trick

In the following, we detail the mathematical arguments leading to the *replica trick* expression (Edwards and Anderson, 1975). For some $r \to 0$, we can write for any scalar x:

$$\begin{aligned} x^r = \exp(r \ln x) &= \lim_{r \to 0} 1 + r \ln x \\ \Rightarrow \lim_{r \to 0} r \ln x &= \lim_{r \to 0} x^r - 1 \\ \Rightarrow \ln x &= \lim_{r \to 0} \frac{x^r - 1}{r} \\ \therefore \mathbb{E}[\ln x] &= \lim_{r \to 0} \frac{\mathbb{E}[x^r] - 1}{r}, \ \mathbb{E} : \text{averaging} \end{aligned} \tag{.2.53}$$

.2.7. Computation of the free-energy

The self-averaged free energy (per unit weight) of our student network, is given by (Engel and Van den Broeck, 2001),

$$-\beta f = \frac{1}{d} \langle \langle \ln Z \rangle \rangle_{z,W} \tag{.2.54}$$

Here, $\beta = 1/T$ is the inverse temperature parameter corresponding to our statistical ensemble, d the (teacher) student network width, and Z the partition function of the system defined as (n: number of training examples).

Leveraging the replica trick, we next obtain,

$$\begin{aligned}\langle\langle Z^r\rangle\rangle_{z,W} &= \prod_{a=1}^{r}\prod_{\mu=1}^{d}\int \mathrm{d}\mu\left(W^a\right)\mathrm{d}y_a^\mu \mathrm{d}(y^*)^\mu e^{-\beta N\mathcal{E}_T(y_a,y^*)} \\ &\times\left\langle\left\langle \delta\left(y^{*\mu}-\frac{1}{\sqrt{d}}W^T x^{*\mu}\right)\delta\left(y_a^\mu-\frac{1}{\sqrt{d}}W_a^T x^\mu\right)\right\rangle\right\rangle_{z,W} \\ &= \prod_{a=1}^{r}\prod_{\mu=1}^{d}\int \mathrm{d}\mu\left(W^a\right)\frac{\mathrm{d}y_a^\mu \mathrm{d}\hat{y}_a^\mu}{2\pi}\frac{\mathrm{d}y^{*\mu}\mathrm{d}\hat{y}^{*\mu}}{2\pi}e^{-\beta N\mathcal{E}_T(y_a,y^*)}e^{iy^{*\mu}\hat{y}^{*\mu}+iy_a^\mu\hat{y}_a^\mu} \\ &\times\left\langle\left\langle \exp\left\{\left(-\frac{i}{\sqrt{d}}\hat{y}^{*\mu}W^T x^{*\mu}-\frac{i}{\sqrt{d}}\hat{y}_a^\mu W_a^T x^\mu\right)\right\}\right\rangle\right\rangle_{z,W}\end{aligned} \tag{.2.55}$$

where in the last line above, we have expressed the inserted δ functions using their integral representations. To make further progress, we introduce the auxiliary variables,

$$\sum_{ija} W_a^i \Delta_{ij} W^{*j} = dR_a, \tag{.2.56}$$

$$\sum_{ij\langle a,b\rangle} W_a^i \Gamma_{ij} W_b^j = dQ_{ab} \tag{.2.57}$$

via the respective δ functions, to arrive at,

$$\begin{aligned}\langle\langle Z^n\rangle\rangle_{z,W} &= \prod_{\mu,a,b}\int \mathrm{d}\mu\left(\mathbf{W}^a\right)\frac{\mathrm{d}y_a^\mu\mathrm{d}\hat{y}_a^\mu}{2\pi}\frac{\mathrm{d}y^{*\mu}\mathrm{d}\hat{y}^{*\mu}}{2\pi}e^{-\beta N\mathcal{E}_T(y_a,y^*)}e^{iy^{*\mu}\hat{y}^{*\mu}+iy_a^\mu\hat{y}_a^\mu} \\ &\times\int P\mathrm{d}Q^{ab}\int P\mathrm{d}R^a\,\delta\left(\sum_{i,j,a}W_a^i\Delta_{i,j}W^{*j}-PR_a\right)\delta\left(\sum_{ij\langle a,b\rangle}W_a^i\Gamma_{ij}W_b^j-PQ_{ab}\right) \\ &\times\left\langle\left\langle \exp\Big(-\frac{Q_0}{2}\sum_{\mu,a}(\hat{y}_a^\mu)^2-\frac{1}{2}\sum_{\mu,\langle a,b\rangle}\hat{y}_a^\mu\hat{y}_b^\mu Q_{ab}-\sum_{\mu,a}\hat{y}^{*\mu}\hat{y}_a^\mu R_a-\frac{1}{2}\sum_\mu(\hat{y}^{*\mu})^2\Big)\right\rangle\right\rangle_W\end{aligned} \tag{.2.58}$$

Repeating the procedure of expressing the above δ functions using their integral representations, we then get ($\alpha = n/d$),

$$\begin{aligned}\langle\langle Z^n\rangle\rangle_{x,x^*,W} = \int \prod_{a,b} \frac{\mathrm{d}Q_0}{\sqrt{2\pi}} \frac{\mathrm{d}\hat{Q}_{0a}}{4\pi} \frac{\mathrm{d}Q_{ab}\hat{Q}_{ab}}{2\pi/d} \frac{\mathrm{d}R_a\hat{R}_a}{2\pi/d} \exp\Big(\frac{iP}{2} \sum_a Q_0\hat{Q}_{0a} + iP \sum_{a<b} Q^{ab}\hat{Q}^{ab} \\ + iP \sum_a R^a\hat{R}^a \Big) \int \prod_{i,a} \frac{\mathrm{d}W_i^a}{\sqrt{2\pi}} \exp\Big(-\frac{i}{2} \sum_{i,j,a} \hat{Q}_{0a} W_a^i \Gamma_{ij} W_a^j \\ - i \sum_{i,j,a<b} \hat{Q}_{ab} W_a^i \Gamma_{ij} W_b^j - i \sum_{i,j,a} \hat{R}_a \Delta_{ij} W_a^j \Big) \times \\ \int \prod_{\mu,a} \frac{\mathrm{d}y_a^\mu \mathrm{d}\hat{y}_a^\mu}{2\pi} \frac{\mathrm{d}y^{*\mu}}{\sqrt{2\pi}} e^{-\beta N \mathcal{E}_T(y_a, y^*)} \exp\Big(-\frac{1}{2} \sum_\mu (y^{*\mu})^2 + i \sum_{\mu,a} \hat{y}_a^\mu \hat{y}_a^\mu \\ - \frac{1}{2} \sum_{a,\mu} \left(1 - R_a^2\right) (\hat{y}_a^\mu)^2 - \frac{1}{2} \sum_{\mu,\langle a,b\rangle} \hat{y}_a^\mu \hat{y}_b^\mu \left(Q^{ab} - R^a R^b\right) - i \sum_{\mu,a} y^{*\mu} \hat{y}_a^\mu R^a \Big)\end{aligned} \tag{.2.59}$$

If we now, perform a singular value decomposition of the covariance matrix Γ as, $\Gamma = \mathrm{U}^T\mathrm{S}\mathrm{U} = \mathrm{V}^T\mathrm{V}$, where S: matrix of singular values of Γ, and we have expressed, $\mathrm{V} = \mathrm{S}^{1/2}\mathrm{U}$, then one can proceed to write,

$$\begin{aligned}\langle\langle Z^n\rangle\rangle_{x,W} = \frac{1}{\det|V|} \int \prod_{a,b} \frac{\mathrm{d}Q_0}{\sqrt{2\pi}} \frac{\mathrm{d}\hat{Q}_{0a}}{4\pi} \frac{\mathrm{d}Q_{ab}\hat{Q}_{ab}}{2\pi/d} \frac{\mathrm{d}R_a\hat{R}_a}{2\pi/d} \exp\Big(\frac{iP}{2} \sum_a Q_0\hat{Q}_{0a} \\ + iP \sum_{a<b} Q^{ab}\hat{Q}^{ab} + iP \sum_a R^a\hat{R}^a \Big) \int \prod_{i,a} \frac{\mathrm{d}\tilde{W}_i^a}{\sqrt{2\pi}} \exp\Big(-\frac{i}{2} \sum_{i,a} \hat{Q}_{0a} \left(\tilde{W}_a^i\right)^2 \\ - i \sum_{i,a<b} \hat{Q}_{ab} \tilde{W}_a^i \tilde{W}_b^i - i \sum_{i,j,a} \hat{R}_a \tilde{W}_a^j \Big) \times \int \prod_{\mu,a} \frac{\mathrm{d}y_a^\mu \mathrm{d}\hat{y}_a^\mu}{2\pi} \frac{\mathrm{d}y^{*\mu}}{\sqrt{2\pi}} e^{-\beta N \mathcal{E}_T(y_a, y^*)} \\ \exp\Big(-\frac{1}{2} \sum_\mu (y_\mu^*)^2 + i \sum_{\mu,a} \hat{y}_a^\mu \hat{y}_a^\mu - \frac{1}{2} \sum_{a,\mu} \left(1 - R_a^2\right) (\hat{y}_a^\mu)^2 - i \sum_{\mu,a} y^{*\mu} \hat{y}_a^\mu R^a \\ - \frac{1}{2} \sum_{\mu,\langle a,b\rangle} \hat{y}_a^\mu \hat{y}_b^\mu \left(Q^{ab} - R^a R^b\right) \Big)\end{aligned} \tag{.2.60}$$

having expressed, $\tilde{W}_a = \mathrm{V}W_a$, and identifying $\Delta = \mathrm{S}^{1/2}\mathrm{U}$ from our definitions. Now, since in the above, the W_i^a integrals factorize in i, and similarly the y_a^μ, $\hat{y}_a^\mu$ and $\mathrm{d}y^{*\mu}$ factorize in μ, one can proceed to write:

$$\begin{aligned}\langle\langle Z^n\rangle\rangle_{x,W} = \frac{1}{\det|V|} \int \prod_{a,b} \frac{\mathrm{d}Q_0 \mathrm{d}\hat{Q}_{0a}}{\sqrt{2\pi}4\pi} \frac{\mathrm{d}Q_{ab}\hat{Q}_{ab}}{2\pi/d} \frac{\mathrm{d}R_a\hat{R}_a}{2\pi/d} \exp\Big(P\Big[\frac{i}{2} \sum_a Q_0\hat{Q}_{0a} \\ + i \sum_{a<b} Q^{ab}\hat{Q}^{ab} + i \sum_a R^a\hat{R}^a + G_S(\hat{Q}_{0a}, \hat{Q}^{ab}, \hat{R}^a) + \alpha G_E(Q^{ab}, R^a) \Big] \Big)\end{aligned} \tag{.2.61}$$

where,

$$G_S(\hat{Q}_{0a},\hat{Q}^{ab},\hat{R}^a) = \ln \int \prod_a \frac{\mathrm{d}\tilde{W}^a}{\sqrt{2\pi}} \exp\Big(-\frac{i}{2}\sum_a \hat{Q}_{0a}\tilde{W}_a^i\tilde{W}_a^i - i\sum_{a<b}\hat{Q}_{ab}\tilde{W}_a\tilde{W}_b - i\sum_a \hat{R}_a\tilde{W}_a\Big)$$

$$G_E(Q^{ab},R^a) = \ln \int \prod_a \frac{\mathrm{d}y_a\mathrm{d}\hat{y}_a}{2\pi}\frac{\mathrm{d}y^*}{\sqrt{2\pi}} e^{-\beta N\mathcal{E}_T(y_a,y^*)} \exp\Big(-\frac{1}{2}(y^*)^2 + i\sum_a \hat{y}_a\hat{y}_a$$
$$-\frac{1}{2}\sum_a\left(1-R_a^2\right)(\hat{y}_a)^2 - \frac{1}{2}\sum_{\langle a,b\rangle}\hat{y}_a\hat{y}_b\left(Q^{ab}-R^aR^b\right) - iy^{*\mu}\sum_a \hat{y}_aR^a\Big) \tag{.2.62}$$

Now, in the limit $d \to \infty$, Eq. .2.61 can be approximated using the saddle-point approach (Bender and Orszag, 2013),

$$\langle\langle Z^n\rangle\rangle_{x,W} \approx \textbf{extr}_{Q_0,\hat{Q}_{0a},Q^{ab},\hat{Q}^{ab},R^a,\hat{R}^a} \exp\Big(P\Big[\frac{i}{2}\sum_a Q_0\hat{Q}_{0a} + i\sum_{a<b}Q^{ab}\hat{Q}^{ab}$$
$$+ i\sum_a R^a\hat{R}^a + G_S(\hat{Q}_{0a},\hat{Q}^{ab},\hat{R}^a) + \alpha G_E(Q^{ab},R^a)\Big]\Big) \tag{.2.63}$$

where, **extr** corresponds to extremization of $\langle\langle Z^n\rangle\rangle_{x,W}$ over the respective order parameters. Performing this extremization over $\hat{Q}_{0a}$, $\hat{Q}^{ab}$ and $\hat{R}^a$, then generates an expression of the form,

$$\langle\langle Z^n\rangle\rangle_{x,W} = \textbf{extr}_{Q_0,Q,R}\exp\Bigg\{nN\Bigg(\frac{1}{2}\frac{Q-R^2}{Q_0-Q} + \frac{1}{2}\ln(Q_0-Q) - \frac{\alpha}{2}\ln\left[1+\beta(Q_0-Q)\right]$$
$$-\frac{\alpha\beta}{2}\frac{1-2R+Q}{1+\beta(Q_0-Q)}\Bigg)\Bigg\} \tag{.2.64}$$

where we have invoked *replica symmetry* in the form, $Q^{ab} = Q$ and $R^a = R$, and that $\mathcal{E}_T = (y^* - y)^2/2$. Plugging this back into Eq. .2.54, then finally yields,

$$\beta f = -\textbf{extr}_{Q_0,Q,R}\Bigg\{\frac{1}{2}\frac{Q-R^2}{Q_0-Q} + \frac{1}{2}\ln(Q_0-Q) - \frac{\alpha}{2}\ln\left[1+\beta(Q_0-Q)\right]$$
$$-\frac{\alpha\beta}{2}\frac{1-2R+Q}{1+\beta(Q_0-Q)}\Bigg\} \tag{.2.65}$$

The remaining pair of order parameters generate the following set of transcendental equations on extremization (Bös, 1998):

$$\begin{aligned} R &= \frac{\alpha}{a} \\ Q &= \frac{\alpha}{a^2-\alpha}\left(1-\frac{2-a}{a}\alpha\right) \\ Q_0 &= Q + \frac{1}{\beta\,(a-1)} \end{aligned} \tag{.2.66}$$

where, $a = \max[1,\alpha]$ for $T \to 0$.

Now, the above determined values of R,Q and Q_0 can be perceived as the *maximally likely* values of R,Q and Q_0 of our teacher-student setup, for an inverse temperature β parameterizing the system.

Supplementary Material For the Second Article

.3. Further discussions

On Primal (parameter space) vs. Dual (feature space) dynamics: Although the cross-entropy loss is convex, it does not admit an analytical solution, even in a simple logistic regression (Xu and Frank, 2004). Importantly, it also does not have a finite solution when the data is linearly separable (Albert and Anderson, 1984) (which is the case in high dimensions (Cover, 1965)). As such, our study is concerned with characterizing the solutions that the training algorithm converges to. A dual optimization approach enables us to describe these solutions in terms of contributions of the training examples (Hsieh et al., 2008). While primal and dual dynamics are not guaranteed to match, the solution they converge to is guaranteed to match (Burges and Crisp, 2000), and that is what our theory builds upon.

For further intuition, we provide a simple experiment in app .5, directly visualizing the primal vs. the dual dynamics as well as the effect of the proposed spectral decoupling method.

The intuition behind Spectral Decoupling (SD): Consider a training datapoint x in the middle of the training process. Intuitively, the model has two options for decreasing the loss of this example:

(1) Get more confident on a feature that has been learned already by other examples. or,

(2) Learn a new feature.

SD, a simple L2 penalty on the output of the work, would favor (2) over (1). The reason is that (2) does not make the network over-confident on previously learned examples, while (1) results in over-confident predictions. Hence, SD encourages learning more features by penalizing confidence. Our principal novel contribution is to characterize this process formally and to theoretically and empirically demonstrate its effectiveness.

From another perspective, here we describe how one can arrive at Spectral Decoupling. From Thm. 6.3.5, we know that Gradient Starvation happens because of the coupling between features (equivalently alphas). We notice that in Eq. 6.3.9, if we get rid of S^2, then the alphas are decoupled. To get rid of S^2 , one can see that instead of $||\boldsymbol{\theta}||^2$ as the regularizer, we should have $||SV^T\boldsymbol{\theta}||^2$. Luckily, this is exactly equal to $||\hat{y}^2||$, since $\hat{y} = \Phi\boldsymbol{\theta} = UV^T\boldsymbol{\theta}$. We would like to highlight that $||SV^T\boldsymbol{\theta}||^2$ as the regularizer means that different directions are penalized according to their strength. It means that we suppress stronger directions more than others which would allow weaker directions

to flourish.

Then why not use Squared-error loss for classification too? The biggest obstacle when using squared-error loss for classification is how to select the target. For example, in a cats vs. dogs classification task, not all cats have the same amount of "catty features". However, recent results favor using squared-error loss for classification and show that models trained with squared-error loss are more robust (Hui and Belkin, 2020). We conjecture that the improved robustness can be attributed to a lack of gradient starvation.

On using NTK: Theoretical analysis of neural networks in their general form is challenging and generally intractable. Neural Tangent Kernel (NTK) has been an important milestone that has simplified theoretical analysis significantly and provides some mechanistic explanations that are applicable in practice. Inevitably, it imposes a set of restrictions; mainly, NTK is only accurate in the limit of large width. Therefore, the common practice is to provide the theoretical analysis in simplified settings and validate the results empirically in more general cases (see, e.g. Huang et al. (2020); Chen et al. (2020b); Wang et al. (2021)). In this work, we build on the same established practices: Our theories analytically study an NTK linearized network; and we further validate our findings on several standard neural networks. In fact, in all of our experiments, learning is done in the regular "rich" (non-NTK) regime, and we verify that our proposed method, as identified analytically, mitigates learning limitations.

Future Directions: This work takes a step towards understanding the reliance of neural networks upon spurious correlations and shortcuts in the dataset. We believe identifying this reliance in sensitive applications is among the next steps for future research directions. That would have a pronounced real-world impact as neural networks have started to be used in many critical applications. As a recent example, we would like to point to an article by researchers at Cambridge (Roberts, 2021) where they study more than 300 papers on detecting whether a patient has COVID or not given their CT Scans. According to the article, none of the papers were able to generalize from one hospital data to another since the models learn to latch on to hospital-specific features. An essential first step is to uncover such reliance and then to design methods such as our proposed spectral decoupling to mitigate the problem.

.4. Experimental Details

.4.1. A Simple Experiment Summarizing the Theory

Here, we provide a simple experiment to study the difference between the primal and dual form dynamics. We also compare the learning dynamics in cases with and without Spectral Decoupling (SD).

Recall that primal dynamics arise from the following optimization,

$$\min_{\theta} \left(\mathbf{1} \cdot \log\left[1 + \exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\left\|\hat{\boldsymbol{\theta}}\right\|^2 \right),$$

while the dual dynamics are the result of another optimization,

$$\max_{\alpha_i} \left[H(\boldsymbol{\alpha}) - \frac{1}{2\lambda} \sum_{jq} \boldsymbol{\alpha}\mathbf{Y}\boldsymbol{\Phi}_0\boldsymbol{\Phi}_0^T\mathbf{Y}^T\boldsymbol{\alpha}^T \right].$$

Also recall that Spectral Decoupling suggests the following optimization,

$$\min_{\theta} \left(\mathbf{1} \cdot \log\left[1 + \exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\left\|\hat{\mathbf{y}}\right\|^2 \right).$$

We conduct experiments on a simple toy classification with two datapoints for which the matrix $\mathbf{U}$ of Eq. 6.3.15 is defined as, $\mathbf{U} = \begin{pmatrix} 0.8 & -0.6 \\ 0.6 & 0.8 \end{pmatrix}$. The corresponding singular values $\mathbf{S} = [s_1, s_2 = 2]$ where $s_1 \in \{2, 3, 4, 5, 6\}$. According to Eq. 6.3.13, when $\mathbf{S} = [2, 2]$, the dynamics decouple while in other cases starvation occurs. Fig. 1 shows the corresponding features of z_1 and z_2. It is evident that by increasing the value of s_1, the value of z_1^* increases while z_2^* decreases (starves). Fig. 1 (left) also compares the difference between the primal and the dual dynamics. Note that although their dynamics are different, they both share the same fixed points. Fig. 1 (right) also shows that Spectral Decoupling (SD) indeed decouples the learning dynamics of z_1 and z_2 and hence increasing the corresponding singular value of one does not affect the other.

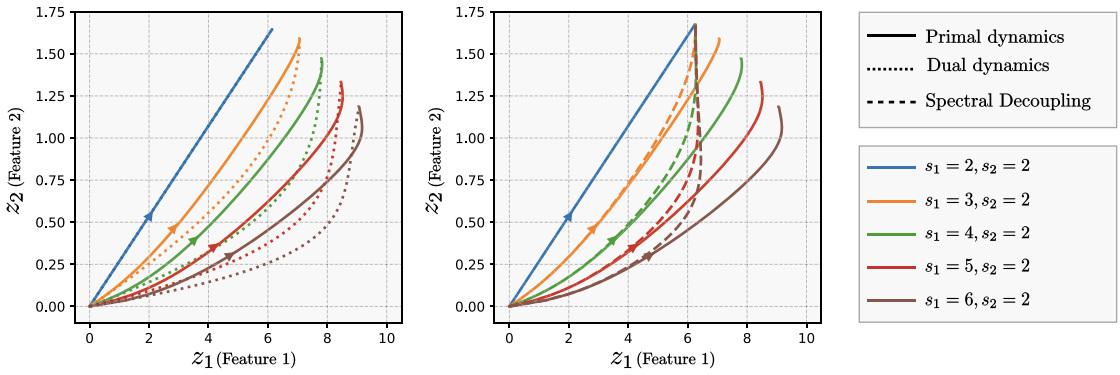


Fig. 1. An illustration of the learning dynamics for a simple 2D classification task. x-axis and y-axis represent learning along features of z_1 and z_2, respectively. Each trajectory corresponds to a combination of the corresponding singular values of s_1 and s_2. It is evident that by increasing the value of s_1, the value of z_1^* increases while z_2^* decreases (starves). **(Left)** compares the difference between the primal and the dual dynamics. Note that although their dynamics are different, they both share the same fixed points. **(Right)** shows that Spectral Decoupling (SD) indeed decouples the learning dynamics of z_1 and z_2 and hence increasing the corresponding singular value of one does not affect the other.

.4.2. Two-Moon Classification: Comparison with other regularization methods

We experiment the Two-moon classification example of the main paper with different regularization techniques. The small margin between the two classes allows the network to achieve a negligible loss by only learning to discriminate along the horizontal axis. However, both axes are relevant for the data distribution, and the only reason why the second dimension is not picked up is the fact that the training data allows the learning to explain the labels with only one feature, overlooking the other. Fig. 2 reveals that common regularization strategies including Weight Decay, Dropout (Srivastava et al., 2014) and Batch Normalization (Ioffe and Szegedy, 2015) do not help achieving a larger margin classifier. Unless states otherwise, all the methods are trained with Full batch Gradient Descent with a learning rate of $1e-2$ and a momentum of 0.9 for $10k$ iterations.

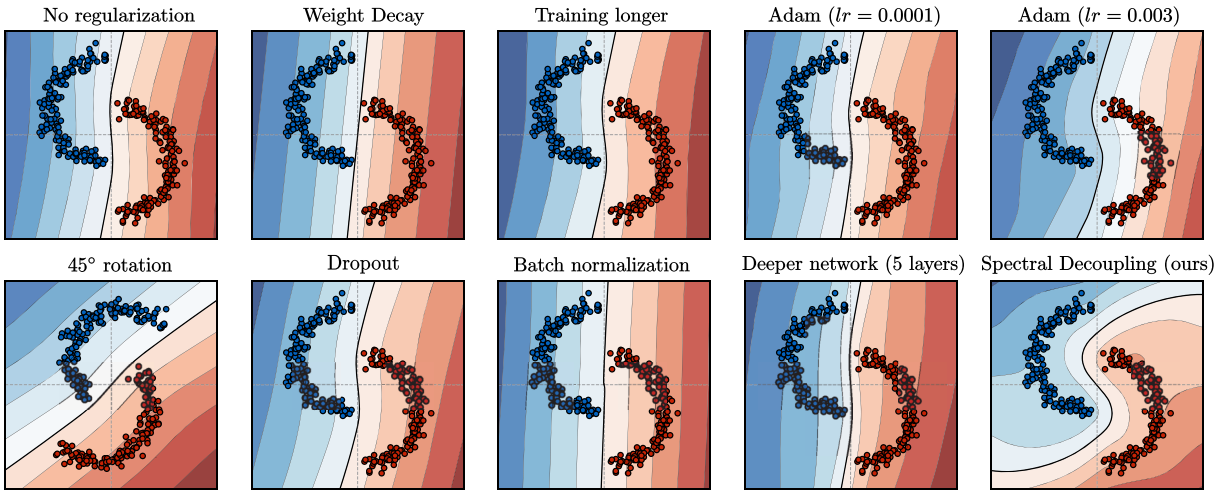


Fig. 2. The effect of common regularization methods on a simple task of two-moon classification. It can be seen that common practices of deep learning seem not to help with learning a curved decision boundary. The acronym "lr" for the Adam optimizer refers to the learning rate. Shown decision boundaries are the average over 10 runs in which datapoints and the model initialization parameters are sampled randomly. Here, only the datapoints of one particular seed are plotted for visual clarity.

.4.3. CIFAR classification

We use a four-layer convolutional network with ReLU non-linearity following the exact setup of Nar et al. (2019). Sweeping λ from 0 to its optimal value results in a smooth transition from green to orange. However, larger values of λ will hurt the IID test (zero perturbation) generalization. The value that we cross-validate on is the average of IID and OOD generalization performance.

.4.4. Colored MNIST with color bias

For the Colored MNIST task, we aggregate all the examples from both training environments. Table. 1 reports the hyper-parameters used for each method.

Method	Layers	Dim	Weight decay	LR	Anneal steps	Penalty coef
ERM	2	300	0.0	1e-4	0/2000	n/a
SD	2	300	0.0	1e-4	450/2000	2e-5
IRM	2	390	0.00110794568	0.00048985365	190/500	91257.18613115903

Table 1. Hyper-parameters used for the Colored-MNIST experiment. Hyper-parameters of IRM are obtained from their released code. "Anneal steps" indicates the number of iterations done before applying the method.

More on Fig. 3. ERM captures the color feature and in the absence of any digit features (environment 4), the network's accuracy is low, as is expected because of reverse color-label match at testing. Moreover, the ERM network is very confident in this environment (confidence is inversely proportional to entropy). The SD network appears to capture the color feature too, with identical classification accuracy in environment 4, but much lower confidence which indicates the other features it expects to classify are absent. Consistent with this, in the case where both color and digit features are present (environment 2), SD achieves significantly better performance than ERM which is fooled by the flipped colors. This is again consistent with SD mitigating GS caused by the color feature onto the digit shape features. Meanwhile, IRM appears to not capture the color feature altogether. Specifically, when only the color is presented to a network trained with IRM, network predicts 50% accuracy with low confidence meaning that IRM is indeed "invariant" to the color as its name suggests. We note that further theoretical justifications are required to fully understand the underlying mechanisms in learning with spurious correlations.

As a final remark, we highlight that, by design, this task assumes access to the test environment for hyperparameter tuning for all the reported methods. This is not a valid assumption in general, and hence the results should be only interpreted as a probe that shows that SD could provide an important level of control over what features are learned.

The hyperparameter search has resulted in applying the SD at 450^{th} step. We observe that 450^{th} step is the step at which the traditional (in-distribution) overfitting occurs. This suggests that one might be able to tune hyperparameters without the need to monitor on the test set.

For all the experiments, we use PyTorch (Paszke et al., 2017). We also use NNGeometry George (2020) for computing NTK.

.4.5. CelebA with gender bias: The experimental details

Figure 4 depicts the learning curves for this task with and without Spectral Decoupling. For the CelebA experiment, we follow the same setup as in Sagawa et al. (2019) and use their released code. We use Adam optimizer for the Spectral Decoupling experiments with a learning rate of $1e-4$ and a batch size of 128. As mentioned in the main text, for this experiment, we use a different variant of

Spectral Decoupling which also provably decouples the learning dynamics,

$$\min_{\theta} \; \mathbf{1} \cdot \left(\log \left[1 + \exp \left(-\mathbf{Y} \hat{\mathbf{y}} \right) \right] + \frac{\lambda}{2} \| \hat{\mathbf{y}} - \gamma \|^2 \right).$$

.4.5.1. Hyper-parameters. We applied a hyper-parameter search on λ and γ for each of the classes separately. Therefore, a total of four hyper-parameters are found. For class zero, $\lambda_0 = 0.088$, $\gamma_0 = 0.44$ and for class one, $\lambda_1 = 0.012$, $\gamma_1 = 2.5$ are found to result in the best worst-group performance.

During the experiments, we found that for the CelebA dataset, classes are imbalanced: 10875 examples for class 0 and 1925 examples for class 1; meaning a ratio of 5.65. That is why we decided to penalize examples of each class separately with different coefficients. We also found that penalizing the outputs' distance to different values γ_0 and γ_1 helps the generalization. As stated in lines 842-844, the hyperparameter search results in the following values: 2.5 and 0.44.

.4.6. Computational Resources

For the experiments and hyper-parameter search an approximate number of 800 GPU-hours has been used. GPUs used for the experiments are NVIDIA-V100 mostly on internal cluster and partly on public cloud clusters.

.5. Proofs of the Theories and Lemmas

.5.1. Eq. 6.3.7 Legendre Transformation

Following Jaakkola and Haussler (1999), we derive the Legendre transformation of the Cross-Entropy (CE) loss function. Here, we reiterate this transformation as following,

Lemma .5.1 (CE's Legendre transformation, adapted from Eq. 46 of Jaakkola and Haussler (1999))**.** *For a variational parameter $\alpha \in [0, 1]$, the following linear lower bound holds for the cross-entropy loss function,*

$$\mathcal{L}(\omega) := \log\Big(1 + e^{-\omega}\Big) \geq H(\alpha) - \alpha\omega, \tag{.5.1}$$

in which $\omega := y\hat{y}$ and $H(\alpha)$ is the Shannon's binary entropy. The equality holds for the critical value of $\alpha^* = -\nabla_\omega \mathcal{L}$, i.e., at the maximum of r.h.s. with respect to α.

PROOF. The **Legendre** transformation converts a function $\mathcal{L}(\omega)$ to another function $g(\alpha)$ of conjugate variables α, $\mathcal{L}(\omega) \to g(\alpha)$. The idea is to find the expression of the tangent line to $\mathcal{L}(\omega)$ at ω_0 which is the first-order Taylor expansion of $\mathcal{L}(\omega)$,

$$t(\omega, \omega_0) = \mathcal{L}(\omega_0) + (\omega - \omega_0) \nabla_\omega \mathcal{L}|_{\omega=\omega_0}, \tag{.5.2}$$

where $t(\omega, \omega_0)$ is the tangent line. According to the Legendre transformation, the function $\mathcal{L}(\omega)$ can be written as a function of the intercepts of tangent lines (where $\omega = 0$). Varying ω_0 along the

x-axis provides us with a general equation, representing the intercept as a function of ω,

$$t(\omega = 0, \omega_0 = \omega) = \mathcal{L}(\omega) - \omega \nabla_\omega \mathcal{L}. \tag{.5.3}$$

The cross-entropy loss function can be rewritten as a soft-plus function,

$$\mathcal{L}(\omega) = -\log \sigma(\omega) = \log\Big(1 + e^{-\omega}\Big), \tag{.5.4}$$

in which $\omega := y\hat{y}$. Letting $\alpha := -\nabla_\omega \mathcal{L} = \sigma(-\omega)$ we have,

$$\omega = \log\left(\frac{1-\alpha}{\alpha}\right), \tag{.5.5}$$

which allows us to re-write the expression for the intercepts as a function of α (denoted by $g(\alpha)$),

$$\begin{aligned} g(\alpha) &= \mathcal{L}(\omega) - \omega \nabla_\omega \mathcal{L} && (.5.6)\\ &= \mathcal{L}(\omega) + \alpha\omega && (.5.7)\\ &= -\alpha \log \alpha - (1-\alpha)\log(1-\alpha) && (.5.8)\\ &= H(\alpha), && (.5.9) \end{aligned}$$

where $H(\alpha)$ is the binary entropy function.

Now, since $\mathcal{L}$ is convex, a tangent line is always a lower bound and therefore at its maximum it touches the original function. Consequently, the original function can be recovered as follows,

$$\mathcal{L}(\omega) = \max_{0 \le \alpha \le 1} \; H(\alpha) - \alpha\omega. \tag{.5.10}$$

Note that the lower bound in Eq. .5.1 is now a linear function of $\omega := y\hat{y}$ but at the expense of an additional maximization over the variational parameter α. An illustration of the lower bound is depicted in Fig. 3. Also a comparison between the dual formulation of other common loss functions is provided in Table. 2.

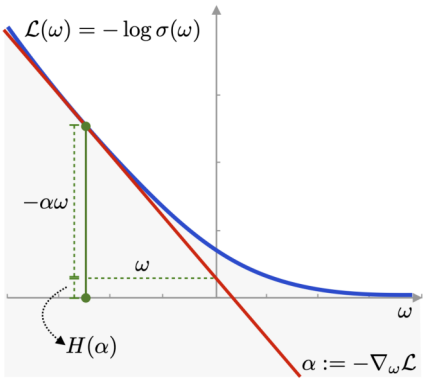


Fig. 3. Diagram illustrating the Legendre transformation of the function $\mathcal{L}(\omega) = \log(1 + e^{-\omega})$. The function is shown in blue, and the tangent line is shown in red. The tangent line is the lower bound of the function: $H(\alpha) - \alpha\omega \leq \mathcal{L}(\omega)$.

Loss	**Primal form**	**Dual form**
Cross-Entropy	$\log(1 + e^{-w})$	$\max_{0<\alpha<1}\big[H(\alpha) - \alpha\omega\big]$
Hinge loss	$\max(0, 1 - w)$	$\max_{0\leq\alpha\leq1}\big[\alpha\mathbf{1}^T - \alpha\omega\big]$
Squared error	$(1 - \omega)^2$	$\max_{\alpha}\big[-\frac{1}{2}\alpha^2 + \alpha - \alpha\omega\big]$

Table 2. Dual forms of other common different loss functions. The dual form of the Hinge loss is commonly used in Support Vector Machine (SVMs). For the ease of notation, we assume scalar ω and α.

□

.5.1.1. Extension to Multi-Class. Building on Eq. 65-71 of Jaakkola and Haussler (1999), which derives the Legendre transform of multi-class cross-entropy, one can update Eq. 6.3.6 of the main paper to

$$-\sum_{c=1}^{C} y_c \log(\hat{y}_c) \geq H(\alpha) - \sum_{c=1}^{C} \alpha^c \hat{y}_c, \tag{.5.11}$$

where $H(\alpha)$ is the entropy function, $C = \#$classes, and vectors of α^c are defined for each class. Then Eq. 6.3.8 of the paper is then updated to,

$$\Big(H(\alpha) - \frac{1}{2\lambda}\sum_{c=1}^{C}(\delta_y - \alpha^c)\Phi\Phi^T(\delta_y - \alpha^c)^T\Big). \tag{.5.12}$$

With a change of variable $\alpha^c := \delta_y - \alpha^c$, the theory of SD should remain unchanged.

.5.2. Eq. 6.3.8 Dual Dynamics

In Eq. 6.3.7, the order of min and max can be swapped as proved in Lemma 3 of Jaakkola and Haussler (1999), leading to,

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\theta) = \max_{\boldsymbol{\alpha}} \min_{\boldsymbol{\theta}} \left(\mathbf{1} \cdot H(\boldsymbol{\alpha}) - \boldsymbol{\alpha}\mathbf{Y}\hat{\mathbf{y}} + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2\right).$$

The solution to the inner optimization is,

$$\boldsymbol{\theta}^{*T} = \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{Y}\boldsymbol{\Phi}_0,$$

which its substitution into Eq. 6.3.7 results in Eq. 6.3.8.

.5.3. Eq. 6.3.9

Simply taking the derivative of Eq. 6.3.8 will result in Eq. 6.3.9. When we introduce continuous gradient ascent, we must define a learning rate parameter. This term is conceptually equivalent to the learning rate in SGD, but in this continuous setting, it has no influence on the fixed point.

.5.4. Eq. 6.3.10 Approximate Dynamics

Approximating dynamics of Eq. 6.3.9 with a first order Taylor expansion around the origin of the second term, we obtain

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \left(\mathbf{S}^2 + \lambda \mathbf{I} \right) \mathbf{U}^T \right).$$

PROOF. Starting from the exact dynamics at Eq. 6.3.9,

$$\dot{\boldsymbol{\alpha}} = \eta \left(-\log \boldsymbol{\alpha} + \log \left(\mathbf{1} - \boldsymbol{\alpha} \right) - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \mathbf{S}^2 \mathbf{U}^T \right), \tag{.5.13}$$

we perform a first-order Taylor approximation of the second term at $\boldsymbol{\alpha} = \mathbf{0}$:

$$\log \left(\mathbf{1} - \boldsymbol{\alpha} \right) = -\boldsymbol{\alpha} + \mathcal{O}\left(\boldsymbol{\alpha}^2 \right). \tag{.5.14}$$

Replacing in Eq. .5.13, we obtain

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \boldsymbol{\alpha} - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \mathbf{S}^2 \mathbf{U}^T \right), \tag{.5.15}$$

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \boldsymbol{\alpha} \mathbf{U} \mathbf{I} \mathbf{U}^T - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \mathbf{S}^2 \mathbf{U}^T \right), \tag{.5.16}$$

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \frac{1}{\lambda} \boldsymbol{\alpha} \mathbf{U} \left(\mathbf{S}^2 + \lambda \mathbf{I} \right) \mathbf{U}^T \right). \tag{.5.17}$$

□

.5.5. Thm. 6.3.3 Attractive Fixed-Points

Theorem .5.2. *Any fixed points of the system in Eq. 6.3.10 is attractive in the domain* $\alpha_i \in (0,1)$.

PROOF. We define

$$f_i(\alpha_j) = \eta \left(\log \frac{(1-\alpha_i)}{\alpha_i} - \frac{1}{\lambda} \sum_{jk} u_{ik} s_k^2 (u^T)_{kj} \alpha_j \right) \tag{.5.18}$$

as the gradient function of the autonomous system Eq. 6.3.9.

We find the character of possible fixed points by linearization. We compute the jacobian of the gradient function evaluated at the fixed point.

$$J_{ik} = \left.\frac{\mathrm{d}f_i(\alpha_j)}{\mathrm{d}\alpha_k}\right|_{\alpha_k^*} \tag{.5.19}$$

$$J_{ik} = \eta\left(-\delta_{ik}\left[\alpha_i^*(1-\alpha_i^*)\right]^{-1} - \lambda^{-1}\sum_l u_{il}s_l^2(u^T)_{lk}\right) \tag{.5.20}$$

The fixed point is an attractor if the jacobian is a negative-definite matrix. The first term is negative-definite matrix while the second term is negative semi-definite matrix. Since the sum of a negative matrix and negative-semi definite matrix is negative-definite, this completes the proof. □

.5.6. Eq. 6.3.11 Feature Response at Fixed-Point

At the fixed point $\boldsymbol{\alpha}^*$, corresponding to the optimum of Eq. 6.3.8, the feature response of the neural network is given by,

$$\mathbf{z}^* = \frac{1}{\lambda}\mathbf{S}^2\mathbf{U}^T\boldsymbol{\alpha}^{*T}.$$

PROOF. The solution to the converged $\boldsymbol{\theta}^*$ at the Fixed-Point $\boldsymbol{\alpha}^*$ of Eq. 6.3.10 is,

$$\boldsymbol{\theta}^{*T} = \frac{1}{\lambda}\boldsymbol{\alpha}^*\mathbf{Y}\boldsymbol{\Phi}_0,$$

which by substitution into Eq. 6.3.4, Eq. 6.3.11 is derived.

□

.5.7. Eq. 6.3.12 Uncoupled Case 1

If the matrix of singular values $\mathbf{S}^2$ is proportional to the identity, the fixed points of Eq. 6.3.10 are given by,

$$\alpha_i^* = \frac{\lambda\mathcal{W}(\lambda^{-1}s^2+1)}{s^2+\lambda}, \qquad z_j^* = \frac{s^2\mathcal{W}(\lambda^{-1}s^2+1)}{s^2+\lambda}\sum_i u_{ij}, \tag{.5.21}$$

where $\mathcal{W}$ is the Lambert W function.

PROOF. When $\mathbf{S}^2 = s^2\mathbf{I}$, Eq. 6.3.10 becomes

$$\dot{\boldsymbol{\alpha}} \approx \eta\left(-\log\boldsymbol{\alpha} - \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{U}\left(s^2+\lambda\right)\mathbf{I}\mathbf{U}^T\right), \tag{.5.22}$$

$$\dot{\boldsymbol{\alpha}} \approx \eta\left(-\log\boldsymbol{\alpha} - \frac{1}{\lambda}\boldsymbol{\alpha}\left(s^2+\lambda\right)\right). \tag{.5.23}$$

Fixed points of this system are obtained when $\dot{\boldsymbol{\alpha}} = 0$:

$$0 = \eta \left(-\log \boldsymbol{\alpha} - \frac{s^2 + \lambda}{\lambda} \boldsymbol{\alpha} \right), \tag{.5.24}$$

$$\log \boldsymbol{\alpha} = -\frac{s^2 + \lambda}{\lambda} \boldsymbol{\alpha}. \tag{.5.25}$$

The solution of this equation is

$$\alpha_i^* = \frac{\lambda \mathcal{W}(\lambda^{-1} s^2 + 1)}{s^2 + \lambda} \tag{.5.26}$$

With $\mathbf{z}$ given by Eq. 6.3.11, we have

$$\mathbf{z} = \frac{s^2 \mathcal{W}(\lambda^{-1} s^2 + 1)}{s^2 + \lambda} \mathbf{U}^T \mathbf{1}, \tag{.5.27}$$

$$z_i^* = \frac{s^2 \mathcal{W}(\lambda^{-1} s^2 + 1)}{s^2 + \lambda} \sum_i u_{ij}. \tag{.5.28}$$

□

.5.8. Eq. 6.3.13 Uncoupled Case 2

If the matrix $\mathbf{U}$ is a permutation matrix, the fixed points of Eq. 6.3.10 are given by,

$$\alpha_i^* = \frac{\lambda \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}, \qquad z_j^* = \frac{s_i^2 \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}. \tag{.5.29}$$

PROOF. When $\mathbf{U}$ is a permutation matrix, it can be made an identity matrix with a meaningless reordering of the class labels . Without loss of generality, we therefore consider $\mathbf{U} = \mathbf{I}$

$$\dot{\boldsymbol{\alpha}} \approx \eta \left(-\log \boldsymbol{\alpha} - \frac{1}{\lambda} \boldsymbol{\alpha} \left(\mathbf{S}^2 + \lambda \right) \mathbf{I} \right). \tag{.5.30}$$

Fixed points of this system are obtained when $\dot{\boldsymbol{\alpha}} = 0$

$$0 = \eta \left(-\log \alpha_i - \frac{1}{\lambda} \alpha_i \left(s_i^2 + \lambda \right) \right), \tag{.5.31}$$

$$\log \alpha_i = -\frac{1}{\lambda} \alpha_i \left(s_i^2 + \lambda \right) \tag{.5.32}$$

The solution of this equation is

$$\alpha_i^* = \frac{\lambda \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}. \tag{.5.33}$$

With $\mathbf{z}$ given by Eq. 6.3.11, we have

$$\mathbf{z}^* = \frac{1}{\lambda} \mathbf{S}^2 \mathbf{I} \boldsymbol{\alpha}^{*T}, \tag{.5.34}$$

$$z_j^* = \frac{s_i^2 \mathcal{W}(\lambda^{-1} s_i^2 + 1)}{s_i^2 + \lambda}. \tag{.5.35}$$

□

.5.9. Lemma 6.3.4 Perturbation Solution

PROOF. Starting from the autonomous system Eq. 6.3.10 and assumption in 6.3.4, we have

$$\boldsymbol{\alpha} = \eta \left(- \log \boldsymbol{\alpha} - \boldsymbol{\alpha} \mathbf{A} \right) \tag{.5.36}$$

where

$$\mathbf{A} = \lambda^{-1} \mathbf{U} (\mathbf{S^2} + \lambda \mathbf{I}) \mathbf{U}^T \tag{.5.37}$$

Since the off-diagonal terms are of order δ, we treat them as a perturbation. The unperturbed system has a solution $\boldsymbol{\alpha}_0$ given by case 2

$$\alpha_{0,i}^* = \frac{\mathcal{W}(A_{ii})}{A_{ii}}. \tag{.5.38}$$

We can linearize the autonomous system Eq. 6.3.10 around the unperturbed solution to find,

$$\dot{\boldsymbol{\alpha}} \simeq \eta \left(- \log \left(\boldsymbol{\alpha}_0^* \right) - \boldsymbol{\alpha}_0^* \odot \operatorname{diag} \left(\boldsymbol{A} \right) + \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}} \left[- \log \left(\boldsymbol{\alpha} \right) - \boldsymbol{\alpha} \odot \operatorname{diag} \left(\boldsymbol{A} \right) \right] \Big|_{\boldsymbol{\alpha}_0^*} (\boldsymbol{\alpha} - \boldsymbol{\alpha}_0^*) \right), \tag{.5.39}$$

$$\dot{\boldsymbol{\alpha}} \simeq \eta \left(1 - \log \left(\boldsymbol{\alpha}^* \right) - \left(\operatorname{diag} \left(\boldsymbol{A} \right) + \boldsymbol{\alpha}_0^{*-1} \right) \odot \boldsymbol{\alpha} \right). \tag{.5.40}$$

We then apply the perturbation given by off-diagonal terms of $\mathbf{A}$ to obtain

$$\dot{\boldsymbol{\alpha}} \simeq \eta \left(1 - \log \left(\boldsymbol{\alpha}^* \right) - \boldsymbol{\alpha} \left[\mathbf{A} + \operatorname{diag} \left(\boldsymbol{\alpha}_0^{*-1} \right) \right] \right), \tag{.5.41}$$

where $\operatorname{diag} \left(\boldsymbol{\alpha}_0^{*-1} \right)$ is the diagonal matrix obtained from $\boldsymbol{\alpha}_0^{*-1}$ and where the inverse is applied element by element.

Solving for $\dot{\boldsymbol{\alpha}} = 0$, we obtain the solution

$$\boldsymbol{\alpha}^* = \left(1 - \log \left(\boldsymbol{\alpha}_0^* \right) \right) \left[\mathbf{A} + \operatorname{diag} \left(\boldsymbol{\alpha}_0^{*-1} \right) \right]^{-1}. \tag{.5.42}$$

□

.5.10. Thm. 6.3.5 Gradient Starvation Regime

Theorem .5.3 (Gradient Starvation Regime)**.** *Consider a neural network in the linear regime, trained under cross-entropy loss for a binary classification task. With definition 6.3.1, assuming coupling between features 1 and 2 as in Eq. 6.3.15 and $s_1^2 > s_2^2$, we have,*

$$\frac{\mathrm{d}z_2^*}{\mathrm{d}s_1^2} < 0, \tag{.5.43}$$

which implies GS.

PROOF. From lemma 6.3.4, and with $\mathbf{U}$ given by Eq. 6.3.15, we find that the perturbatory solution for the fixed point is

$$\alpha_1^* = \left[\lambda\left(W\left(\frac{\lambda+\delta^2\left(s_1^2-s_2^2\right)+s_2^2}{\lambda}\right)+1\right)\left(\delta\sqrt{1-\delta^2}\left(s_2^2-s_1^2\right)+\lambda e^{W\left(\frac{\lambda+\delta^2\left(s_1^2-s_2^2\right)+s_2^2}{\lambda}\right)}\right.\right.$$
$$\left.\left.\left(W\left(\frac{\lambda+\delta^2\left(-s_1^2\right)+\delta^2 s_2^2+s_1^2}{\lambda}\right)+1\right)\right)\right]\delta^2\left(\delta^2-1\right)\left(s_1^2-s_2^2\right)^2+$$
$$\left(\lambda+\delta^2\left(s_1^2-s_2^2\right)+\lambda e^{W\left(\frac{\lambda+\delta^2\left(s_1^2-s_2^2\right)+s_2^2}{\lambda}\right)}+s_2^2\right)$$
$$\left.\left(\lambda+\delta^2 s_2^2-\left(\delta^2-1\right)s_1^2+\lambda e^{W\left(\frac{\lambda+\delta^2\left(-s_1^2\right)+\delta^2 s_2^2+s_1^2}{\lambda}\right)}\right)\right]^{-1}$$

$$\alpha_2^* = \left[\lambda\left(W\left(\frac{\lambda+\delta^2\left(-s_1^2\right)+\delta^2 s_2^2+s_1^2}{\lambda}\right)+1\right)\left(\delta\sqrt{1-\delta^2}\left(s_2^2-s_1^2\right)+\lambda e^{W\left(\frac{\lambda+\delta^2\left(-s_1^2\right)+\delta^2 s_2^2+s_1^2}{\lambda}\right)}\right.\right.$$
$$\left.\left.\left(W\left(\frac{\lambda+\delta^2\left(s_1^2-s_2^2\right)+s_2^2}{\lambda}\right)+1\right)\right)\right]$$
$$\left[\delta^2\left(\delta^2-1\right)\left(s_1^2-s_2^2\right)^2+\left(\lambda+\delta^2\left(s_1^2-s_2^2\right)+\lambda e^{W\left(\frac{\lambda+\delta^2\left(s_1^2-s_2^2\right)+s_2^2}{\lambda}\right)}+s_2^2\right)\right.$$
$$\left.\left(\lambda+\delta^2 s_2^2-\left(\delta^2-1\right)s_1^2+\lambda e^{W\left(\frac{\lambda+\delta^2\left(-s_1^2\right)+\delta^2 s_2^2+s_1^2}{\lambda}\right)}\right)\right]^{-1}$$

We have found at Eq. 6.3.11 that the corresponding steady-state feature response is given by

$$\mathbf{z}^* = \frac{1}{\lambda}\mathbf{S}^2\mathbf{U}^T\boldsymbol{\alpha}^{*T} \tag{.5.44}$$

In the perturbatory regime δ is taken to be a small parameter. We therefore perform a first-order Taylor series expansion of $\mathbf{z}^*$ around $\delta = 0$ to obtain

$$z_1^* = \frac{\delta s_1^2\left(W\left(\frac{\lambda+s_2^2}{\lambda}\right)+1\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_2^2\right)}{\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_1^2\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_2^2\right)}+\frac{\lambda s_1^2 e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}\left(W\left(\frac{\lambda+s_1^2}{\lambda}\right)+1\right)\left(W\left(\frac{\lambda+s_2^2}{\lambda}\right)+1\right)}{\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_1^2\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_2^2\right)} \tag{.5.45}$$

$$z_2^* = \frac{\lambda s_2^2 e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}\left(W\left(\frac{\lambda+s_1^2}{\lambda}\right)+1\right)\left(W\left(\frac{\lambda+s_2^2}{\lambda}\right)+1\right)}{\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_1^2\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_2^2\right)}-\frac{\delta s_2^2\left(W\left(\frac{\lambda+s_1^2}{\lambda}\right)+1\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_1^2\right)}{\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_1^2\right)\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_2^2\right)} \tag{.5.46}$$

Taking the derivative of z_2^* with respect to s_1, we find

$$\frac{\mathrm{d}z_2^*}{\mathrm{d}s_1^2} = -\frac{\delta\lambda s_2^2\left(e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)} - e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}\right)\left(W\left(\frac{\lambda+s_1^2}{\lambda}\right)+1\right)}{\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_1^2}{\lambda}\right)}+s_1^2\right)^2\left(\lambda+\lambda e^{W\left(\frac{\lambda+s_2^2}{\lambda}\right)}+s_2^2\right)} \tag{.5.47}$$

Knowing that the exponential of the W Lambert function is a strictly increasing function and that $s_1^2 > s_2^2$, we find

$$\frac{\mathrm{d}z_2^*}{\mathrm{d}s_1^2} < 0. \tag{.5.48}$$

□

.5.11. Eq. 6.3.18 Spectral Decoupling

SD replaces the general L2 weight decay term in Eq. 6.3.5 with an L2 penalty exclusively on the network's logits, yielding

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbf{1}\cdot\log\left[1+\exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\|\hat{\mathbf{y}}\|^2.$$

The loss can be written as,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \mathbf{1}\cdot\log\left[1+\exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\|\boldsymbol{\Phi\theta}\|^2,\\ &= \mathbf{1}\cdot\log\left[1+\exp\left(-\mathbf{Y}\hat{\mathbf{y}}\right)\right] + \frac{\lambda}{2}\left\|\boldsymbol{SV}^T\boldsymbol{\theta}\right\|^2,\\ &\geq \mathbf{1}\cdot\left[H(\boldsymbol{\alpha}) - \boldsymbol{\alpha Y}\hat{\mathbf{y}}\right] + \frac{\lambda}{2}\left\|\boldsymbol{SV}^T\boldsymbol{\theta}\right\|^2.\end{aligned}$$

Optimizing $\mathcal{L}(\boldsymbol{\theta})$ wrt to $\boldsymbol{\theta}$ results in the following optimum,

$$\boldsymbol{\theta}^{*T} = \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{Y}\boldsymbol{\Phi}_0\boldsymbol{VS}^{-2}\boldsymbol{V}^T,$$

which by substitution into the loss function, the dynamics of gradient ascent leads to,

$$\dot{\boldsymbol{\alpha}} = \eta\left(\log\frac{\mathbf{1}-\boldsymbol{\alpha}}{\boldsymbol{\alpha}} - \frac{1}{\lambda}\boldsymbol{\alpha}\mathbf{U}\mathbf{S}^2\mathbf{S}^{-2}\mathbf{U}^T\right) = \eta\left(\log\frac{\mathbf{1}-\boldsymbol{\alpha}}{\boldsymbol{\alpha}} - \frac{1}{\lambda}\boldsymbol{\alpha}\right),$$

where $\log$ and division are taken element-wise on the coordinates of $\boldsymbol{\alpha}$ and hence dynamics of each α_i is independent of other $\alpha_{j\neq i}$.