Deep Networks Generalization and Optimization Trajectory

Stanisław Jastrzębski

Jagiellonian University

Faculty of Mathematics and Informatics

Supervisors:

Prof. Jacek Tabor (Jagiellonian University)
Prof. Amos Storkey (University of Edinburgh)

Cracow 2018

Contents

Summary

Deep Learning (DL) has led to numerous breakthroughs in fields such as computer vision or natural language processing. However, theoretical reasons behind this success remain still unclear. For example, the number of parameters of a deep network routinely exceeds by orders of magnitude the number of training examples. This is somewhat in contrast with the intuition that simpler models should generalize better (i.e. work better on unseen examples); as John von Neumann said: "*With four parameters I can fit an elephant, and with five I can make him wiggle his trunk*". This apparent tension between the success of DL and lack of understanding of its theoretical underpinnings motivates this work.

The approach we take starts from the observation that *the common denominator of most modern deep networks is that they are trained using stochastic gradient descent (SGD)*, a simple optimization algorithm. Previous work has argued that optimization acts as an implicit regularizer. Depending on the optimization algorithm, or its hyperparameters, the final solution can have different generalization properties. We build on this line of thought.

This thesis is composed of six papers. To summarize, we consider as the key contribution the set of empirical and theoretical results about SGD–based training of deep networks. We highlight here three main take-aways:

- Curvature of the loss surface is highly influenced by the learning rate and the batch size (hyperparameters of SGD); both during training and at the final minima.
- The scale of stochastic noise in SGD is controlled by the learning rate to batch size ratio, and is a crucial factor for the learning dynamics and the final generalization performance.
- After the early phase of training both the complexity of the function in the input space and curvature of the loss surface are largely determined.

Practically speaking, results in this thesis can help practitioners train neural networks; for instance by simplifying finding the optimal hyperparameters in SGD. Besides, we hope our work will help develop in the future new optimizers tailor-fit to deep neural networks.

Streszczenie

Głębokie Uczenie (ang. *Deep Learning*, DL) pozwala osiągnąć rewolucyjne wyniki w takich dziedzinach jak wizja komputerowa, czy przetwarzanie języka naturalnego. Niestety, teoretyczne własności sieci neuronowych (modele rozważane w DL) które stoją za tym sukcesem są dalej niejasne. Na przykład sieci neuronowe mają często o rzęd wielkości więcej parametrów niż rozmiar zbioru trenującego. Jest to sprzeczne z intuicją, że proste modele powinny generalizować lepiej (tj. działać lepiej na nowych przykładach). Ta pozorna sprzeczność motywuje badanie teoretycznych podstaw sieci neuronowych.

Naszym punktem startowym jest obserwacja, że większość głębokich sieci neuronowych jest trenowana z użyciem prostego algorytmu Stochastic Gradient Descent (SGD). Wcześniejsze prace pokazują, że wybór metod optymalizacji wpływa na generalizację modelu. W ramach tej pracy rozwiniemy ten nurt badań.

Na pracę doktorską składa się sześć publikacji. Głównym rezultatem jest zestaw teoretycznych i empirycznych wyników pogłębiających nasze zrozumienie trenowania sieci z użyciem SGD. Podsumowywując najważniejsze z nich to:

- Na krzywiznę funkcji kosztu ma duży wpływ krok uczenia i rozmiar próbki (hiperparametry SGD). Nie tylko w końcowym minimum, ale też w czasie całego trenowania.
- Skala szumu stochastycznego w SGD jest kontrolowana przez iloraz kroku uczenia do rozmiaru próbki. Ten iloraz jest kluczowym czynnikiem wpywającym na dynamikę uczenia i końcową generalizację.
- Po początkowej fazie uczenia poziom skomplikowania funkcji (w przestrzeni wejścia) i krzywizna funkcji kosztu są w dużej mierze ustalone.

Z praktycznego punktu widzenia wyniki w tej rozprawie mogą pomóc praktykom trenować sieci neuronowe, przykładowo upraszczając dobór hiperparametrów SGD. Mamy także nadzieję, że nasza praca przyczyni się w przyszłości do rozwoju optymalizatorów dostosowanych do sieci neuronowych.

Acknowledgements

Foremost, I would like to thank my family; in particular, my parents for their continuous support. I thank my fiancé for her love and understanding, which she had in spite of my long working hours.

Most of the research presented in this thesis was carried out during my stays at the University of Edinburgh, and at the University of Montreal. I am greatly indebted to my collaborators, especially to: Devansh Arpit, Dzmitry Bahdanau, Nicolas Ballas, Asja Fischer, Arian Hosseini, Zac Kenton, David Krueger, and Michael Nouvkovitz. I also thank my collaborators from Jagiellonian University: Wojciech Czarnecki, Damian Leśniak, Igor Podolak, and Igor Sieradzki.

As someone said, everyone in academia is smart, but not everyone is kind. I am very grateful for the opportunity I had to work with Prof. Yoshua Bengio.

Last but not least, I wanted to thank my supervisors for these three years. In particular, Prof. Jacek Tabor for being open to advising a PhD student in deep learning[1], and Prof. Amos Storkey for his truly unique approach to science.

[1] I am sorry I have never derived that variant of SVM.

Chapter 1

Introduction

Deep learning (DL) is a subfield of Artificial Intelligence (AI) concerned with learning hierarchical models called neural networks [23]. Deep learning methods have recently led to breakthrough results in traditional subfields of AI such as computer vision [34], natural language processing [6] or speech recognition [44].

Despite these successes, theoretical reasons behind neural networks success remain still unclear. For example, deep networks are usually heavily over-parametrized, i.e. the number of network parameters routinely exceeds by orders of magnitude the number of training examples, as shown in Fig. 1.1. This is somewhat in contrast with the intuition that smaller models should generalize better; as John von Neumann said: "With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.". See also Fig. 1.2.

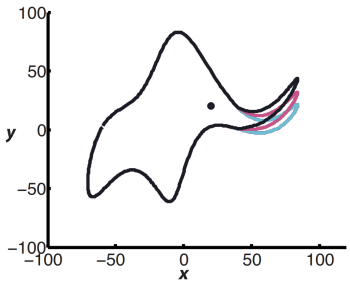


Figure 1.2: A function parametrized by 5 parameters that resembles an elephant. One of the parameters can modulate (*wiggle*) trunk of the elephant. Reproduced from [43].

The way in which neural networks generalize to unseen examples is often counterintuitive. Deep models are surprisingly *brittle* when contrasted with the way humans make decisions. An illustrative example is that it is possible for almost any neural network to find a small perturbation of the input called an *adversarial perturbation* that is imperceptible to human, but changes prediction of the model [22]. These generalization issues even led some researchers to proposing that a large paradigm shift will be required to achieve the next level of performance in AI [41].

This apparent tension between success of deep learning and lack of understanding of its theoretical underpinnings is the general motivation behind this thesis. Our high-level goal is to provide novel perspectives on why deep networks generalize.

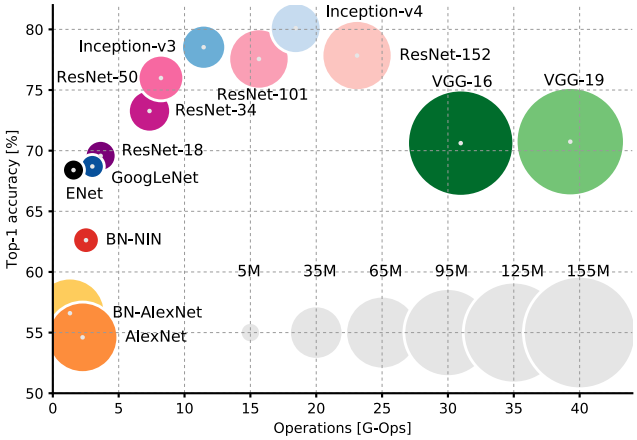


Figure 1.1: Deep neural networks become increasingly more over-parametrized. Reproduced from [13].

The approach we take starts from the observation that the common denominator of most modern DNNs is that they are trained using stochastic gradient descent (SGD), a simple optimization algorithm. Previous work, such as [48], has argued that optimization acts as an *implicit regularizer*. Depending on the choice of the optimization algorithm, or its hyperparameters, the final solution will have different generalization properties.

Building on this line of work we investigate how some, previously identified, *correlates of generalization* are influenced by the optimization algorithm; in other words, instead of understanding directly the cause of generalization, we will try to understand evolution along the *optimization trajectory* of quantities that have been shown to correlate with generalization. Our main research goal is to tackle the following open research question:

What is the role of optimization trajectory
in explaining generalization properties of deep networks?

Our investigation is supplemented by complementary studies on the role of architecture and representation.

1.1 Structure

This thesis is composed of the following papers (P_1) - (P_6) (main authorship is indicated by an underline):

(P_1) S. Jastrzębski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey. Width of Minima Reached by Stochastic Gradient Descent is Influenced by Learn-

ing Rate to Batch Size Ratio. *Artificial Neural Networks and Machine Learning – ICANN 2018*. Lecture Notes in Computer Science 392–402. 2018.

(P_2) D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, S. Lacoste-Julien, A Closer Look at Memorization in Deep Networks. *Proceedings of the 34th International Conference on Machine Learning*. PMLR 70:233-242. 2017.

(P_3) S. Jastrzębski, Z. Kenton, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey. DNN's Sharpest Directions Along the SGD Trajectory. *Modern Trends in Nonconvex Optimization for Machine Learning Workshop at International Conference on Machine Learning*. 2018.

(P_4) S. Jastrzębski, D. Arpit, N. Ballas, V. Verma, T. Che, Y. Bengio. Residual Connections Encourage Iterative Inference. *International Conference on Learning Algorithms (Conference Track)*. 2017.

(P_5) S. Jastrzębski, D. Bahdanau, S. Hosseini, M. Noukhovitch, Y. Bengio, J. C. K. Cheung. Commonsense Mining as Knowledge Base Completion? A Study on the Impact of Novelty. *Proceedings of the Workshop on Generalization in the Age of Deep Learning*, 8-16. 2018.

(P_6) S. Jastrzębski, D. Leśniak, W. M. Czarnecki. Learning to SMILE(S). *International Conference on Learning Algorithms (Workshop Track)*. 2016.

In the Appendix we include extended versions of (P_1) and (P_3), which we encourage the reader to consult (under the changed titles *"Three Factors Influencing Minima in SGD"* and *"On the Relation of DNN Loss and SGD Step Length"*). We would like to especially highlight (P_3) which has been substantially extended and is currently under review.

The rest of this thesis is structured as follows. In Ch. 2 we provide the necessary background, which we will later expand on as necessary in the following sections. The papers are introduced in the two independent chapters: Ch. 3 and Ch. 4. In these chapters we generally shift focus to the contributions that were more directly influenced by the author of this thesis. Each paper is introduced in a separate section, and where applicable supplemented with a discussion of follow-up work.

In Ch. 3 we describe the main results of this thesis based on (P_1), (P_2) and (P_3). In these papers we investigate the links between the optimization trajectory and generalization of neural networks. The three papers that were authored during two research visits in Prof. Yoshua Bengio's group, and done in a collaboration with my co-supervisor Prof. Amos Storkey.

In Ch. 4 we introduce complementary work published as (P_4), (P_5) and (P_6). These papers give some additional insights about generalization in the context of natural language processing and computer assisted drug design.

We conclude in Sec. 5. We discuss there progress made on explaining generalization properties of deep networks, and the open research directions.

Chapter 2

Background

Understanding the link between the final generalization and different aspects of neural networks is an active research topic. In this short chapter we will introduce the necessary background, as well as generally motivate studying generalization properties of neural networks. In Sec. 2.1 we will provide a high level introduction to supervised learning and optimization from the perspective of deep networks. In Sec. 2.2 we will discuss generalization aspects of deep networks and highlight scenarios under which deep networks succeed or fail to generalize.

2.1 Supervised learning and optimization in a nutshell

On the whole, in this thesis we focus on *supervised learning* problems. In this setting the algorithm is provided with N training points consisting of inputs and labels $\{(\mathbf{x}_i, y_i)\}$, $i = 1 \dots N$. Typically, $\mathbf{x}_i$ is a D dimensional vector (e.g. an image) and y_i is a scalar (e.g. correct class of the image). The goal of supervised learning is to *generalize*, i.e. to make *good* predictions on unseen examples.

More formally, let $\mathcal{L}(\hat{y}_i, y_i; \boldsymbol{\theta}) \in \mathbb{R}$ denote *loss function*, where $\hat{y}_i$ and y_i are typically scalars denoting prediction of the model and target, and $\boldsymbol{\theta}$ are the parameters defining the model. Next, let $\mathcal{P}(\mathbf{x}, y)$ denote the joint probability of the unseen inputs and labels. Now we can define the goal of supervised as identifying the parameters $\boldsymbol{\theta}$ achieving minimum loss function $\mathcal{L}$ on the unseen distribution $\mathcal{P}(\mathbf{x}, y)$:

$$\mathcal{L}(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}}[\mathcal{L}(\hat{y}(\mathbf{x}, \boldsymbol{\theta}), y)], \tag{2.1}$$

where we slightly overloaded the notation to denote dependence of the prediction on the input $\mathbf{x}$ and the parameter vector $\boldsymbol{\theta}$. While there are many popular choices of loss functions,

in this thesis we focus on the cross-entropy loss, which is given by

$$\mathcal{L}^{CE}(\hat{y}, y) = \sum_{k=1}^{K} y_k \log \hat{y}_k, \tag{2.2}$$

where K is the number of classes. The cross-entropy loss is amenable to optimization techniques, while at the same time upper-bounds the misclassification rate.

Unfortunately, in most practical scenarios we do not have access to $\mathcal{P}(\mathbf{x}, y)$, and have to resort to using a finite sample. As we will see DNNs achieve good generalization in cases that training examples closely match the training distribution, but often fail to generalize in situations in which there is some sort of a misalignment between the two distributions.

Next, let us briefly introduce optimization of neural networks – the key topic of this thesis. The goal of optimization can be formalized as finding minimum of the loss function $\mathcal{L}$ computed on the training set, i.e. the objective is to find θ^*:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\hat{y_i}(\mathbf{x_i}; \boldsymbol{\theta}), y_i). \tag{2.3}$$

We would like to stress that the overarching goal of supervised learning is to identify a well generalizing solution. Finding a solution achieving low *training error*, while failing to generalize is an undesired behavior referred to as *overfitting*.

Stochastic Gradient Descent (SGD) is one of the most widely used algorithms to approximately solve Eq. 2.3. SGD iteratively minimizes loss by stepping in the negative direction of the estimated gradient of the loss as:

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \eta \hat{g}^S(\boldsymbol{\theta}(t)), \tag{2.4}$$

where η is a scalar called *learning rate*, and $\boldsymbol{\theta}(t)$ denotes parameters at time t, and $\hat{g}^S$ is the gradient approximation. The gradient is approximated using a random subset of S examples called the *mini-batch* as $\hat{g}^S(\boldsymbol{\theta}(t)) = \frac{1}{S} \sum_{i=1}^{S} \nabla \mathcal{L}(x_i, y_i; \boldsymbol{\theta}(t))$. Parameters at $t = 0$ are typically initialized to some random values.

Stochastic gradient descend is successful in optimizing many modern deep networks [23]. This might be considered surprising, because SGD is a greedy algorithm. For example, consider that it is easy to construct a single dimensional loss surface with two minima that differ in height, such that SGD would get stuck in the suboptimal one. Another potential shortcoming of SGD is that it does not adapt to the curvature of the loss surface; it is again easy to construct adversarial scenarios under which SGD would not perform well. Many studies tried to elucidate what makes DNNs amenable to SGD–based training. In particular over-parametrization of neural networks is commonly believed to be an important factor [4].

Given the widespread popularity, success, and simplicity of SGD, in this thesis we will narrow down our focus to this algorithm. Nevertheless, most of our conclusions should

carry over in some form to other optimization algorithms.

2.2 Explaining and characterizing generalization

In certain cases DNNs generalize extremely well compared to other models, such as kernel machines. First widely acclaimed breakthrough using deep networks of Krizhevsky et al. [34] demonstrated a large neural network to improve dramatically upon the previous state of the art which used a manually defined set of features. Spectacular results were also achieved in other domains, such as speech recognition [44], and machine language translation [6].

Despite the aforementioned successes, DNNs tend to fail to generalize in a more broader sense. We can find many examples of such failures even in the tenant of deep networks: the large scale image classification. In an illustrative recent study Azulay and Weiss [5] it is shown that DCNNs *are not* robust to image translations, which might be surprising because this is the key motivatio behind their design. Image classification datasets contain many biases (e.g. eyes of a dog or person will be usually in the center of an image) and DCNNs leverages them to make predictions.

The existence of *adversarial examples* is yet another example of a surprising generalization failure of DNNs. In its simplest form, an adversarial example is a perturbed genuine example in such a way that (i) the perturbation is imperceptible to human, and (ii) prediction of a DNN on this example is changed. Most neural networks are sensitive to such adversarial perturbations [22].

The aim of this thesis is to contribute to understanding of these successes and failures of deep networks. On the whole, many less and more formal approaches to analysing generalization properties of deep networks exist. To start with, architecture of DNNs is believed to be the key factor contributing to their success [35, 53]. In particular, Krizhevsky et al. [34] used a deep convolutional neural network (DCNNs), which process an image by applying the same operation to multiple locations. This is tailor fit to image classification, as typically location of the given feature in the image is not critical for making the prediction. It seems that success of deep learning would not be possible without the introduction of such key architectural blocks; another example is the long-short term memory architecture [28], which is routinely used to process sequential data.

Perhaps a more general view is that DNNs are biased towards automatically discovering *good* representations [10]. An illustrative example is that typically DCNNs learn to *detect* edges in the first layers[1]; the subsequent layers can be shown to detect higher level features, such as parts of objects. In a nutshell, deep neural networks are naturally biased towards

[1] One common approach to analyze function of a given neuron is to find input pattern maximally activating it [23].

learning a well generalizing mapping from the input space to the output space due to their architectural choices.

On the more theoretical side, *statistical learning theory* [55] is a classical theoretical framework for studying generalization. In particular, it proposes various bounds on the error on the unseen examples, typically using the training error and a *complexity measure* of the model. For instance, complexity measures using the raw number of parameters are commonly used for model selection [2].

Classical complexity measures have been shown to correlate poorly with generalization performance of neural networks [59]. The key reason is that DNNs tend to be heavily over-parametrized, in which case these standard complexity measures tend to give overly pessimistic upper bounds. One way to reconcile this is to understand better role of the optimization process, which is able to find a well generalizing solution in this over-parametrized model space. For instance, Murata et al. [47] investigated the role of optimization process and proved a mathematical relation between the optimization hyperparameters, curvature of the final minima, and the final generalization performance.

While there are many complementary approaches to studying generalization properties of neural networks, in the main part of this thesis, Ch. 3, we will build on work along the lines of [47] and focus on the role of optimization. Later, in Ch. 4 we will provide complementary results from the architecture and representation point of view.

Chapter 3

Generalization and Optimization Trajectory

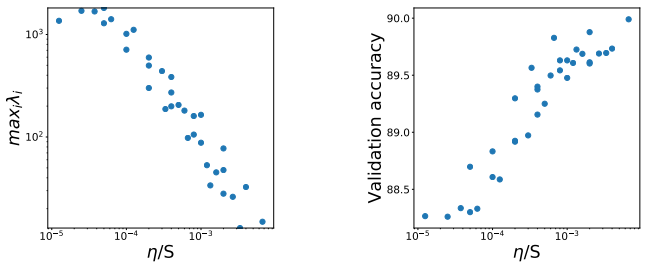
This chapter introduces our core contributions, published as (P_1), (P_2), and (P_3). On a high level, the main goal of these papers is to provide novel evidence supporting the importance of optimization process for determining the final generalization performance.

More concretely, we aim to improve our understanding of the relationship between generalization performance of neural networks and the *optimization trajectory* they take in the loss surface, during training. By the optimization trajectory we will refer to *characteristics of neural network exhibited on the training set at different moments during optimization*. In particular, we will focus on the curvature of the loss surface, and the so-called *memorization* phenomena. In each paper, (P_1) to (P_3), the overarching goal will be to link these characteristics to the final generalization performance. We also highlight that we will take a closer look at the optimization process in the initial phase of training, which is rarely studied in the literature.

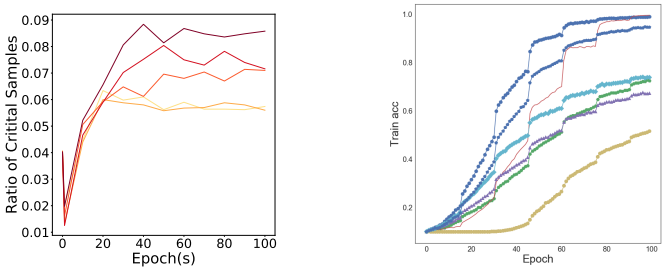
Amongst other prior work, this chapter is largely motivated by Zhang et al. [59], which demonstrates that neural network are able to learn an arbitrary labeling of the training data. This implies that the loss surface of DNNs has multiple *trivial* minima. At the same time neural networks trained on real data are able to generalize to unseen examples. *This motivates our study of how SGD steers away from such trivial minima.*

Our contributions to the state of the art are the following:

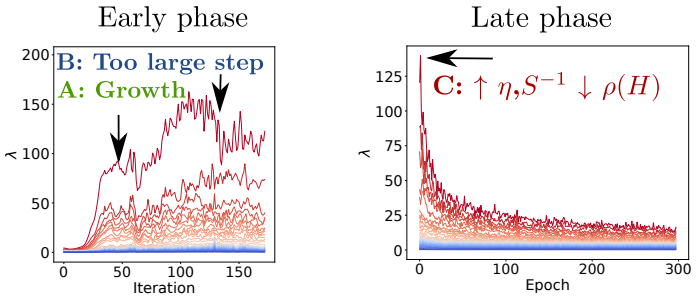
- In Sec. 3.2 extending prior work we demonstrate that not only the batch size used in SGD, but also the learning rate influences curvature of the final minima. See also Fig. 3.1a.
- In Sec. 3.2 we provide a novel theoretical argument that learning rate and batch size have an exchangeable role. We suggest importance of the learning rate to batch size ratio in determining the dynamical and convergent properties of SGD. See also



(a) Figures reproduced from (P_1). SGD runs corresponding to the same ratio of learning rate η to batch size S achieve similar learning dynamics and curvature of the final endpoint. **Left**: spectral norm of the Hessian evaluated at the final endpoint (y axis) corresponding to SGD with different ratio of learning rate to batch size (x axis). **Right**: validation accuracy (y axis) corresponding to SGD with different learning rate to batch size ratios (x axis).



(b) Figures reproduced from (P_3). **Left**: Complexity of the model (y axis, evaluated using the Critical Sample Ratio measure) stabilizes after first epoch of training, and is larger for models trained on larger amount of noise (color). **Right**: Training speed on random labels is significantly reduced by common regularizers. such as dropout. Each curve represents accuracy (y axis) under a different regularizer (color) over epochs of training (x axis).



(c) Figures reproduced from (P_2). Schematic illustration of the evolution of the loss surface along the sharpest direction during training. Curvature along this direction initially grows (A). In most iterations, we find the loss surface along the sharpest direction to resemble a bowl, such that SGD crosses its *minima*, often taking *a too large step* (B and C). Finally, curvature stabilizes or decays with a peak value determined by learning rate and batch size (C).

Figure 3.1: Selected contributions from (P_1), (P_2) and (P_3).

Fig. 3.1a.

- In Sec. 3.3 we demonstrate a link between memorization and robustness in the input space. We observe that a larger degree of memorization correlates with a larger sensitivity to perturbations in the input space. See also Fig. 3.1b.
- In Sec. 3.3 and Sec. 3.4 we characterize behavior of SGD in the early phase – typically the first few epochs of training. We find that SGD finds a flat region of the loss surface with curvature dependent on learning rate and batch size. Furthermore, SGD steers away from memorization in the early phase. See also Fig. 3.1b and Fig. 3.1c.

3.1 Related work

Before diving into our contributions, we briefly discuss here the most related prior work that have largely inspired (P_1) to (P_3). Please also refer to Ch. 2 for a more general discussion.

3.1.1 Motivating studies

Our study is generally motivated by the wide-spread use of SGD in neural network community and the practical importance of setting optimally the mini-batch size S and the learning rate η of SGD. Nowadays, virtually all neural networks are trained using variants of SGD. The learning rate is often claimed to be the most important hyperparameter in training deep networks [23]. Similarly, using a small batch size has been widely observed to correlate with good generalization, e.g. in [31, 45].

A simple, but thought-provoking investigation that has inspired a large part of this chapter as we briefly mentioned was carried out in Zhang et al. [59]. The key finding in Zhang et al. [59] is that deep networks when provided with examples that had randomized labels can learn them efficiently, which is called *memorization*. It is important to emphasize that this capability to *memorize* data is not specific to DNNs. A related phenomena can be demonstrated for any machine learning model, provided it has a sufficient capability [8].

A key implication of these results is that for any labeling of the training labels there exists a minima in the loss surface such that the network has *memorized* these examples and has no generalization power. The key question is why SGD *steers away* from such a minima and ends in one associated with a good generalization performance. Note that this question can be easily answered in the context of linear regression and gradient descent. Advani and Saxe [1] points to the simple fact that gradient descent focuses on directions with largest curvature first, which is argued to correspond to learning a function that does not memorize the training set.

Another key investigation was carried out in [27, 31]. It was first proposed in [27] that a so-called *flat* minima should exhibit a good performance on unseen examples. Flatness of the minima can be measured by the volume it occupies in the weight space, which under the quadratic approximation can be computed using the Hessian evaluated at the parameters. Keskar et al. [31] demonstrated empirically that SGD using a large batch size generally converged to a sharp minima associated with bad generalization. Importantly, this connects geometry of the loss surface and the final generalization performance, on which we build in this chapter.

However, we want to stress that the correlation between generalization and curvature of the final minima might not be causal. Recent work in [17] has demonstrated that due to the existence of many symmetries in typical neural networks it is possible through a *reparametrization* of the parameter space to change sharpness of the minima, while maintaining its generalization properties. Nevertheless, this empirical correlation strongly motivated our research.

3.1.2 Selected prior work on SGD

Understanding the reasons behind SGD effectiveness in finding well-generalizing solutions remains an important open research question. The angle we will take in this paper is centered on understanding the loss surface and properties of SGD. Here we will briefly introduce the key investigations in these areas.

Deep neural networks are routinely over-parametrized; these models typically use substantially more parameters than the size of the training set. Based on this it is natural to be concerned that depending on the initialization the optimization could get stuck in a suboptimal local minima; however, it seems that these bad local minima are not of a practical importance [30, 21]. This apparent paradox is partially reconciled in Goodfellow et al. [21], which demonstrated that the loss surface of DNNs resembles a smooth convex function along the line connecting the initialization point and the final minimizer. This view is further developed in Kleinberg et al. [32], where it is shown formally that the loss surface of DNNs is *one-point* convex, which intuitively means that after smoothing (using the noise induced by SGD) the loss function resembles a convex function along the slice connecting the current iterate and the final minima.

Despite these and other fortunate properties, the loss surface of a deep network can be notoriously difficult to optimize using SGD. One of the key challenges is that the loss surface can be highly ill-conditioned, meaning that some directions in the weight space can have a disproportionate curvature compared to others [49]. This restricts the possible learning rates to low values, which empirically hurts both the optimization speed and worsens the final generalization. Many solutions to this problem have been proposed. In particular, one of the most successful and widely used modules in the design of deep networks is Batch Normalization [29], which as recent work argues dramatically improves conditioning of

the loss surface [50].

Finally, let us discuss some of the prior work on understanding the noise structure of SGD which is relevant to findings in (P_1) and (P_2). We will follow the same notation as in Sec. 2.1.

One popular approach to model SGD noise structure is to assume that the difference between the true gradient $g(\boldsymbol{\theta}$ and its approximation using the mini-batch $\hat{g}^S(\boldsymbol{\theta})$ is a normally distributed variable with 0 mean and variance $C^S(\boldsymbol{\theta})$. Then, taking the SGD step can be interpreted as taking a step using the true gradient perturbed by a normally distributed random variable:

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \eta(g(\boldsymbol{\theta}) + \mathcal{N}(0, C^S(\boldsymbol{\theta})), \tag{3.1}$$

where the scale of noise depends on the batch size as $C^S(\boldsymbol{\theta}) = \frac{1}{S}C(\boldsymbol{\theta})$.

From the simple analysis above it might seems that only the batch size affects the scale of the stochastic noise. However, as argued in [12, 33] a similar dynamics of learning is maintained when the *ratio* of learning rate and batch size is kept constant. Bottou et al. [11] provides first theoretical arguments for importance of this ratio, but only in a strongly convex setting. We will extend these results theoretically to DNNs in (P_1), and study further the practical importance of learning rate to batch size ratio in (P_1) and (P_3).

3.2 Width of Minima Reached by SGD is Influenced by Learning Rate to Batch Size Ratio (P_1)

In the first paper of this chapter, (P_1), we investigate how hyperparameters of SGD, the learning rate η and the batch size S, influence properties of the final endpoint of optimization, as well as the entire optimization trajectory.

There are two main contributions of (P_1) that we want to highlight here. First, we give a novel theoretical argument for the linear dependence of stochastic noise induced by SGD on the ratio of learning rate and batch size. In particular, we argue that the same ratio of learning rate to batch size results in a similar learning dynamic of training DNNs. To achieve this following [37] we approximate SGD by a Stochastic Differential Equation (SDE) and draw attention to the fact that the stochastic noise is scaled by $\frac{\eta}{S}$ in the SDE which approximates SGD:

$$d\vec{\theta} = -\mathbf{g}(\vec{\theta})dt + \sqrt{\frac{\eta}{S}}\mathbf{R}(\vec{\theta})d\vec{W}(t) \ . \tag{3.2}$$

The second contribution is characterizing dependence of the curvature of the final minima on learning rate and batch size used during training. For a minima with the corresponding

Hessian $\mathbf{H}$ and the expected loss $\mathbb{E}(L)$, under certain assumptions, we derive the following relation:

$$\mathbb{E}(L) = \frac{\eta}{4S}\mathrm{Tr}(\mathbf{H}), \tag{3.3}$$

where L and $\mathbf{H}$ are loss and Hessian of this loss at the final minima. The key implication is that assuming the same height, a higher ratio of learning rate to batch size will steer SGD towards a wider minima, which generalizes empirical results from [31]. The theoretical argument is further supported by an empirical investigation on standard neural networks. See also Fig. 3.1a.

In the extended version of (P_1)[1] we further investigate importance of the learning rate to batch size ratio. First, we show its positive impact on the final robustness of the model to memorization. Finally, we demonstrated that decreasing learning rate during training can be replaced by an equivalent linear increase of batch size. This last result is particularly relevant to practitioners that have access to large computing resources, in which case using large batch sizes is desirable.

To the best of our knowledge (P_1) was the first work to demonstrate dependence of curvature of the final endpoint on the learning rate used, as well as the theoretical importance of the learning rate to batch size ratio in controlling the stochastic noise scale via the $g = \sqrt{\frac{\eta}{S}}$ factor[2]. In the following we discuss some of the work citing (P_1)[3].

Some of the follow-up work investigated regimes in which learning rate and batch size are not exchangeable [58, 42]. We note that the extended version of (P_1) already states that the theory is applicable when the batch size is close to the size of the training set. This is further confirmed by [42]; one observation therein is that small batch sizes lead to more stable learning dynamics and permit a larger maximum learning rate than the linear relation would predict.

Finally, in (P_1) we focused on the convergent properties of SGD, making just few comments on how learning rate and batch size impact the overall learning dynamics. This aspect was studied by [58, 61, 7]. In particular, Baity-Jesi et al. [7] investigates the differences between the late and the early phase of training. It is identified that at the end of training the dynamics resemble closely a stationary diffusion, in agreement with some of the assumptions used in deriving Eq. 3.3.

[1] Please refer to the Appendix. Alternatively, it is accessible online at `https://arxiv.org/abs/1711.04623`.

[2] Concurrently with [52, 14].

[3] According to Google Scholar (P_1) is cited 19 times (accessed 10.2018).

3.3 A Closer Look At Memorization in Deep Networks (P_2)

In this section we introduce the second paper, (P_2), in which we extend the prior work of Zhang et al. [59]. The key open question of Zhang et al. [59] which we tackle is how DNNs are able to learn a meaningful function, instead of simply *memorizing* the training set (see Sec. 3.1 for a more detailed discussion).

Let us focus here on selected contributions of (P_2). The first contribution we want to highlight is proposing a measure of complexity called *critical sample ratio* (CSR) that quantifies robustness of the network to perturbation in the input space. We empirically demonstrate that a model memorizing examples achieves a higher CSR. Further, in (P_2) we track input space robustness throughout the training. We observe that complexity of the model is largely determined in the early phase of training, typically increasing within the first epochs of training. This is suggestive of the view that models learn first simple *patterns* from the data, before memorizing individual examples.

Second, we show in (P_2) that standard regularization techniques, such as dropout or weight decay, limit the speed at which network memorizes the perturbed portion of the training data, while not hindering learning from the *clean* examples. See also Fig. 3.1b.

Finally, let us discuss selected follow-up work of (P_2)[4]; we will focus on the papers citing (P_2) in the context of the discussed contributions. Regularization impact on memorization was further studied in [60, 46, 18]. [60, 46] confirms identified in (P_2) impact of using dropout regularization on the memorization behaviour. Zhang et al. [60] demonstrated that the developed there in augmentation method *mix-up* improves upon dropout in term of slowing down memorization.

The link between susceptibility to adversarial examples and the memorization phenomena was further explored in Ma et al. [40] and Valle Pérez et al. [54]. Ma et al. [40] studies to this end a new metric called "local intrinsic dimensionality". Both papers use the introduced in (P_2) CSR metric to quantify complexity of the model.

3.4 DNN's Sharpest Directions Along SGD Trajectory (P_3)

Extending and connecting (P_1) and (P_2) in (P_3) we focus on the curvature of the loss surface throughout SGD-based training of DNNs. The joint message of (P_2) and (P_3) is that *the early phase of training is critical in determining properties of the final endpoint*, such as curvature of the loss surface (studied in (P_3)) or complexity of the final model (studied in (P_2)). A substantially extended version of (P_3), which is currently under review, is included in the Appendix.

[4] According to Google Scholar (P_2) is cited 60 times (accessed 10.2018).

The main novelty of (P_3) is performing an in-depth empirical investigation of the curvature of the loss surface, along the optimization trajectory. We demonstrate that *curvature of the loss surface throughout the training is a function of the used learning rate and batch size*. In particular, SGD using a large learning rate or a small batch size locates a flat region of the loss surface early on in the training, typically within the first few epochs.

Another key novelty of (P_3) is the observation that SGD follows distinct learning dynamics along the sharp and the flat directions in the loss surface. Immediately after the early phase training it seems that SGD *oscillates* in the subspace corresponding to the sharpest direction, while successfully descending along other directions. In particular we investigated a variant of SGD called *Nudged* SGD, which was demonstrated to be easily steerable towards flatter or sharper regions of the loss surface, while maintaining generalization performance. As such, Nudged SGD provides empirical evidence that curvature of the final minima does not fully determine generalization. See also Fig. 3.1c.

3.5 Discussion

Inspired by the recent studies such as [59] and [31] we looked at the memorization phenomena and curvature of the loss surface along the optimization trajectory, with the end goal of improving our understanding of the role of SGD in improving the final generalization.

The main outcome is a set of theoretical and empirical results about the process of optimizing neural networks. In (P_1) we have studied the relation between SGD hyperparameters, dynamics of learning, and properties of the final minima. We found that the learning rate to batch size ratio is a key determinant of the overall behavior of SGD. Further, in (P_2) and (P_3) we have identified importance of the early phase of training. This rarely studied phase has a large impact on the final solution; deep networks initially dramatically increase their complexity and determine curvature of the loss surface. In summary, we have improved the previous state of the art by considering the link between optimization trajectory and generalization.

These results are not only of a theoretical value; many of them are rather practical. The exchangeable role of learning rate and batch size discussed in (P_1) suggests ways to reduce the hyperparameter space of SGD. Varying batch size schedules, which we discussed in the extended version of (P_1), offer potential for better use of large computing resources. Memorization phenomena which we characterized in (P_2) is not only an academic concept. On the contrary, learning a DNN on any dataset with noisy labels will be affected by this phenomena. Finally, results in (P_3) show that vanilla SGD-based training is inefficient and unstable in the sense that SGD largely oscillates along the sharpest direction in the DNN loss surface (for a more detailed discussion please see also the extended version of (P_3)).

However, there still remains much to be done before we truly understand how SGD helps find well generalizing solutions. For instance, in (P_3) we show empirically that the curva-

ture of the region that SGD converges to in certains situations is not predictive of generalization (which was suggested by the prior work such as [31]). Our investigations suggest several directions for future work:

- We provide ample further evidence that SGD is a crucial factor contributing to the good generalization properties of neural networks. However, it remains unclear if using SGD is the optimal way to regularize training of neural networks.
- The connection between curvature of the loss surface identified in (P_1) and (P_3) and memorization discussed in (P_2) and [59] is yet to be better understood.
- Convergence speed of SGD seems to be largely dependent on the learning rate and batch size used [51]. However, the reason for this is not apparent from the theory that we have developed.
- Developing better theoretical understanding of scenarios in which the learning rate is not exchangeable with the batch-size [42] would be interesting.

Chapter 4

Connecting Representation, Architecture and Generalization

In this chapter we describe complementary studies published as (P_4), (P_5) and (P_6) in which we explore the importance of architecture and representation in explaining generalization properties of neural networks. In (P_4) we focus on a specific connection between architecture and bias exhibited in *residual neural networks*. In (P_5) and (P_6) we focus on two challenging applications of deep networks, acquiring commonsense knowledge and virtual screening, and tackle generalization challenges therein. These challenging tasks have actually motivated a large portion of the studies discussed in Ch. 3. Each section introduces a separate paper. Some additional background is provided at beginning of each section.

4.1 Residual Connections Encourage Iterative Inference (P_4)

In the first paper of this chapter we investigate a link between a widely successful neural architecture called *residual neural networks* (ResNets) [26] and optimization. The key idea behind ResNets is to *residually* transform input; each residual block transforms an input $\mathbf{h}^L$ to $\mathbf{h}^{L+1}$ as $\mathbf{h}^{L+1} = \mathbf{h} + \mathcal{F}(\mathbf{h}^L; \boldsymbol{\theta})$ for some operator $\mathcal{F}$ parametrized by parameters $\boldsymbol{\theta}$. ResNets are composed of multiple stacked residual blocks.

Our work extends Greff et al. [24], which based on several pieces of evidence argues that residual networks perform *iterative estimation*, which is defined informally therein as an incremental processing of the representation towards a given target. In contrast to Greff et al. [24] in (P_4) we study iterative estimation formally. We find that blocks in residual layers are implicitly encouraged towards moving representations in the direction that minimizes a notion of a local to the block loss, which as we argue resembles gradient descent

on a loss surface.

Follow-up work[1] has provided further arguments supporting the iterative inference interpretation of residual networks, and proposed variants more explicitly designed to perform a form of iterative computation [15, 36]. Han et al. [25] has demonstrated a similar bias towards an incremental reduction of a notion of a local loss in the case of a different neural architecture.

4.2 Commonsense mining as knowledge base completion? A study on the impact of novelty (P_5)

The next paper that we wish to introduce deals with the problem of acquiring commonsense knowledge. As we briefly discussed in Sec. 2.2, commonsense knowledge is an umbrella term for the collection of facts about the everyday world. For instance, consider a photo of a man who saws off a tree branch on which he sits. Humans immediately and intuitively understand implications of this man's actions. In contrast, neural networks are often criticised for failing to deal with problems that require such commonsense knowledge. For example Marcus [41] claims that most deep models have problems answering questions such as *"Who is taller, Prince William or his baby son Prince George?"*. In particular, lack of commonsense knowledge can lead to a severe lack of generalization.

Endowing natural language processing systems with commonsense knowledge is one of the key long term goals of the natural language processing field, and a very active research topic [16]. One particular problem is extracting commonsense knowledge from raw text, which we refer to as *commonsense mining*. An approach to this problem has been proposed in Li et al. [38].

In (P_5) motivated by the importance of attaining commonsense knowledge we analyse the approach proposed by Li et al. [38]. We provide empirical evidence that the proposed system is not capable of performing commonsense mining. A trivial solution is found, which in essence memorizes the commonsense statements from the training set. We conclude the paper with practical suggestions on improving evaluation of commonsense mining. The proposed evaluation was recently used in [3].

4.3 Learning to SMILE(s) (P_6)

The last paper of this chapter deals with Computer Assisted Drug Design (CADD). CADD is a multi-step process using computational method to aid discovery of new drugs. A popular analogy for the CADD process is that the goal of CADD is to find a key (a molecule) to

[1] According to Google Scholar (P_4) is cited 9 times (accessed 10.2018).

match a given keyhole (target, commonly a binding site of a protein). Typically, one of the first stages in the CADD pipeline is *virtual screening*. Virtual screening uses computational methods to select a small set of molecules from a large set of candidates.

Our interest in virtual screening stems from generalization challenges that arise when applying neural networks to this problem [56]. The key objective of virtual screening process is to propose a small set of *structurally novel* compounds that are promising candidates for a new drug. As we have discussed in Ch. 2.2 neural networks often have issues generalizing to novel examples, and virtual screening is a particularly good testbed for understanding key reasons behind these challenges.

Motivated by these challenges, we investigate importance of learning representations instead of using a handcrafted set of features. In (P_6) we consider the problem of representing chemical compounds for use in neural networks. Previous methods have largely used handcrafted representation, such as Molprint2D [9]. Instead, we propose a simple approach of using a textual representation called SMILES. Using SMILES allows one to easily apply standard sequential neural networks. It is important to stress that models which use as input directly the molecule graph (instead of its textual representation) are likely to outperform methods using SMILES. The appeal of the method is in its simplicity, and sometimes competitive results.

To the best of our knowledge (P_6) was the first to propose using the SMILES textual representation of compounds in neural networks. This somewhat straightforward idea has been concurrently proposed in [20] and due to its simplicity became popular[2]. Currently, this approach is used in a large body of work in cheminformatics, e.g. [39, 19]. In particular, results in the MoleculeNet benchmark [57][3] indicate that in many standard tasks a model using SMILES as input is competitive to more complex models. However, it was not yet demonstrated how the representation, including ours, impacts the out-of-distribution generalization in virtual screening, which we hope to address in future work.

4.4 Discussion

The take home message of (P_4), (P_5) and (P_6) in the context of this thesis is that generalization cannot be explained by just looking at a single factor. In (P_4) we found that architecture of residual network adds a somewhat surprising bias towards performing iterative inference. In (P_5) we found that a previously proposed model fails to truly generalize in the commonsense mining task. A similar lack of generalization has been observed in virtual screening, and our study in (P_6) on representation is a preliminary attempt at this problem. Interestingly, to this date both of the tasks that we discussed suffer from these generalization issues and arguably only preliminary attempts have been made to fix them.

[2] According to Google Scholar (P_6) is cited 12 times (accessed 10.2018).

[3] We note that (P_6) is not cited in this paper.

This might suggest that a large improvement in the state of the art is needed across the board to improve performance of deep networks in these and other similarly difficult tasks.

Chapter 5

Conclusions

The main goal of this thesis was to investigate what is the role of optimization trajectory in explaining generalization properties of deep neural networks. To this end, in (P_1)–(P_3) we have studied optimization trajectory of SGD–based training of modern deep networks. We consider as the key contribution of this thesis the set of empirical and theoretical results published in these papers, and we highlight here three main take-aways:

- Curvature of the loss surface is highly influenced by the learning rate and the batch size. Both at the final minima and throughout the training.
- The scale of stochastic noise in SGD is controlled by the learning rate to batch size ratio, and is crucial in determining the learning dynamics and the final generalization of SGD.
- After the early phase of training both the complexity of the function in the input space, as well curvature of the loss surface in the weight space, are largely determined.

Besides studies on optimization, we have made efforts towards understanding and improving generalization from other perspectives in (P_4)–(P_6). We hope that in the future a better understanding of the relation between optimization and architecture or representation will emerge.

Overall, it seems clear that optimization and generalization *cannot* be studied separately in deep learning. Yet, current optimizers are typically motivated from the perspective of classical optimization theory. Our parting thought to the reader is that *the relation between optimization trajectory of DNNs and generalization should be reflected in the design of future optimizers.* We hope that our work will provide building blocks for developing such optimizers[1].

[1] Actually, some of our preeliminary results already point towards this direction.

Bibliography

[1] M. S. Advani and A. M. Saxe. High-dimensional dynamics of generalization error in neural networks. *ArXiv e-prints*, October 2017.

[2] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[3] Anonymous. Do language models have common sense? In *Submitted to International Conference on Learning Representations*, 2019. under review.

[4] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *ArXiv e-prints*, February, 2018.

[5] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *ArXiv e-prints*, May, 2018.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ArXiv e-prints*, September, 2014.

[7] M. Baity-Jesi, L. Sagun, M. Geiger, S. Spigler, G. Ben Arous, C. Cammarota, Y. LeCun, M. Wyart, and G. Biroli. Comparing Dynamics: Deep Neural Networks versus Glassy Systems. *ArXiv e-prints*, March 2018.

[8] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 540–548, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[9] Andreas Bender, Hamse Y. Mussa, Robert C. Glen, and Stephan Reiling. Similarity searching of chemical databases using atom environment descriptors (MOLPRINT 2d): Evaluation of performance. *Journal of Chemical Information and Modeling*, 44 (5):1708–1718, 2004.

[10] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *ArXiv e-prints*, June 2012.

[11] L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[12] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

[13] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *ArXiv e-prints*, May, 2016.

[14] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *ArXiv e-prints*, abs/1710.11029, 2017.

[15] Marco Ciccone, Marco Gallieri, Jonathan Masci, Christian Osendorfer, and Faustino J. Gomez. Nais-net: Stable deep networks from non-autonomous differential equations. *ArXiv e-prints*, abs/1804.07209, 2018.

[16] Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103, 9 2015. ISSN 0001-0782.

[17] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. *International Conference on Learning Representations*, 2017.

[18] Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 1919–1925, 2017.

[19] G. B. Goh, N. O. Hodas, C. Siegel, and A. Vishnu. SMILES2Vec: An Interpretable General-Purpose Deep Neural Network for Predicting Chemical Properties. *ArXiv e-prints*, December 2017.

[20] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[21] I. J. Goodfellow, O. Vinyals, and A. M. Saxe. Qualitatively characterizing neural network optimization problems. *ArXiv e-prints*, December 2014.

[22] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[23] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[24] Klaus Greff, Rupesh Kumar Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation. *International Conference on Learning Representations*, 2016.

[25] Kuan Han, Haiguang Wen, Yizhen Zhang, Di Fu, Eugenio Culurciello, and Zhongming Liu. Deep predictive coding network with local recurrent processing for object recognition. *ArXiv e-prints*, abs/1805.07526, 2018.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV (4)*, volume 9908 of *Lecture Notes in Computer Science*, pages 630–645. Springer, 2016.

[27] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1): 1–42, 1997.

[28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667.

[29] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[30] Kenji Kawaguchi. Deep Learning without Poor Local Minima. *Nips*, (Nips):586–594, 2016. ISSN 10495258.

[31] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations*, 2016.

[32] Robert Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does SGD escape local minima? *ArXiv e-prints*, abs/1802.06175, 2018.

[33] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *ArXiv e-prints*, April, 2014.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[36] Sam Leroux, Pavlo Molchanov, Pieter Simoens, Bart Dhoedt, Thomas Breuel, and Jan Kautz. Iamnn: Iterative and adaptive mobile neural network for efficient image classification. *ArXiv e-prints*, April, 2018.

[37] Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and adaptive stochastic gradient algorithms. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of

Proceedings of Machine Learning Research, pages 2101–2110, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[38] Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. Commonsense knowledge base completion. In *Proc. of ACL*, 2016.

[39] B. Liu, B. Ramsundar, P. Kawthekar, J. Shi, J. Gomes, Q. Luu Nguyen, S. Ho, J. Sloane, P. Wender, and V. Pande. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ArXiv e-prints*, June 2017.

[40] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. M. Erfani, S.-T. Xia, S. Wijewickrema, and J. Bailey. Dimensionality-Driven Learning with Noisy Labels. *ArXiv e-prints*, June 2018.

[41] Gary Marcus. Deep learning: A critical appraisal. *ArXiv e-prints*, January, 2018.

[42] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *ArXiv e-prints*, abs/1804.07612, 2018.

[43] Jürgen Mayer, Khaled Khairy, and Jonathon Howard. Drawing an elephant with four complex parameters. *American Journal of Physics*, 78(6):648–649, June 2010.

[44] Tomas Mikolov, Anoop Deoras, Daniel Povey, Lukás Burget, and Jan Cernocký. Strategies for training large scale neural network language models. In David Nahamoo and Michael Picheny, editors, *ASRU*, pages 196–201. IEEE, 2011.

[45] Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors. *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*. Springer, 2012.

[46] A. S. Morcos, D. G. T. Barrett, N. C. Rabinowitz, and M. Botvinick. On the importance of single directions for generalization. *ArXiv e-prints*, March 2018.

[47] Noboru Murata, Shuji Yoshizawa, and Shun ichi Amari. Network information criterion-determining the number of hidden units for an artificial neural network model. *IEEE transactions on neural networks*, 5 6:865–72, 1994.

[48] Behnam Neyshabur. Implicit regularization in deep learning. *ArXiv e-prints*, September, 2017.

[49] Levent Sagun, Utku Evci, V. Ugur Güney, Yann Dauphin, and Léon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *ArXiv e-prints*, abs/1706.04454, 2017.

[50] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How Does Batch Normalization Help Optimization? (No, It Is Not About Internal Covariate Shift). *ArXiv e-prints*, May 2018.

[51] L. N. Smith. Cyclical Learning Rates for Training Neural Networks. *ArXiv e-prints*, June 2015.

[52] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. Don't decay the learning rate, increase the batch size. *ArXiv e-prints*, abs/1711.00489, 2017.

[53] G. Urban, K. J. Geras, S. Ebrahimi Kahou, O. Aslan, S. Wang, R. Caruana, A. Mohamed, M. Philipose, and M. Richardson. Do Deep Convolutional Nets Really Need to be Deep and Convolutional? *ArXiv e-prints*, March 2016.

[54] G. Valle Pérez, C. Q. Camargo, and A. A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *ArXiv e-prints*, May 2018.

[55] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[56] I. Wallach and A. Heifets. Most Ligand-Based Classification Benchmarks Reward Memorization Rather than Generalization. *ArXiv e-prints*, June 2017.

[57] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande. MoleculeNet: A Benchmark for Molecular Machine Learning. *ArXiv e-prints*, March 2017.

[58] C. Xing, D. Arpit, C. Tsirigotis, and Y. Bengio. A Walk with SGD. *ArXiv e-prints*, February 2018.

[59] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *International Conference in Learning Representations*, 2016.

[60] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond Empirical Risk Minimization. *International Conference in Learning Representations*, 2018.

[61] Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma. The Anisotropic Noise in Stochastic Gradient Descent: Its Behavior of Escaping from Minima and Regularization Effects. *ArXiv e-prints*, February 2018.