Séminaire TURING – ENS Paris-Saclay

Why does Deep Learning work so well ? An attempt at an explanation.

Sacha ELKOUBI Even MATENCIO Eustache LE BIHAN Paul SITOLEUX

January 12, 2024

Contents

1 Introduction

Neural networks have been tremendously successful, and deep learning has become standard for tackling a wide range of problems, with the main fields of application being image and natural language processing, but also providing spectacular advances in specialised scientific fields, like protein shape prediction. Despite their impressive success, there remains gaping holes in the understanding of *how* and *why* they work.

The first question can be partially answered within the framework of statistics and classical machine learning: they learn a bio-inspired, hierarchical representation of the data and approximate the targeted function over this representation. Thanks to hardware availability, and the application of an array of tricks motivated by classical machine learning or general intuition, they can do this in an efficient way. Nevertheless, these representation are not easily intelligible by humans and require a significant amount investigations. Also, it appears that deep learning models follow some patterns remaining quite mysterious.

Answering the second question requires to throw grandma's statistical intuition out the window, as assessed in sec.3. In classical machine learning, over-parameterized models do not generalize due to overfitting. In classical Machine Learning, generalization bounds allow to predict model performance for a given data set size, and, one can obtain probabilistic bounds on the generalization error using complexity measures such as the Rademacher complexity and the Vapnik–Chervonenkis (VC) dimension (3.1). For deep models, this breaks down as over-parameterized models often generalizing better than simpler models. Deep Double Descent (3.2) illustrates this phenomenon.

Different hypothesis regarding these "unconventional" behaviors are explored in 4. Specifically, we focus on the way data is processed in deep learning models (see 4.1), and the different structures identified in neural networks that may account for their good generalization properties (4.2.1). We also give insights into a specific aspect of neural networks training, namely Singular Learning Theory 5.1, and we slightly clarify the visualisation one should has of neural networks learning 5.2.

2 Are neural networks truly mimicking the human brain ?

Before diving into more technical considerations we start discussing the biological inspirations of the deep learning. Neural networks, as their name hints, are inspired by observations of the cerebral cortex. The perceptron [12], introduced in the 1950s, aimed at replicating the massively parallel architectures observed in brains. Convolutional Neural Networks (CNNs) further integrate established structures from

the lower visual processing areas in primates. For this reason, a first naive idea to assess neural networks performance is to scrutinize their architecture. At first glance, it would not be surprising if an algorithm mimicking human brain reasoning structures were to achieve human-class results on tasks such as image classification.

Indeed, it has been shown that the activities in layers of CNNs tend to mirror those found in the corresponding human lower and higher visual processing regions [4]. Yaoda Xu and al. [15] confirmed evidence of correspondences between CNNs and the human brain in lower-level (contours, edges, angles, etc) visual representation of real-world objects, yet it is noteworthy that CNNs fall short in fully capturing higher-level (semantic features, on top of low-level features) visual representations of both real-world and artificial objects.

Moreover, recent studies [14] have demonstrated that the hippocampus, a crucial brain structure for memory function, essentially operates to some extent as transformers. When coupled with recurrent positional encoding, transformers can replicate neural representations identified in both the cortex and hippocampus. Notably, transformers seem to closely align with the existing mathematical model of the hippocampus. While attributing the remarkable efficiency of transformers solely to these similarities would be premature and likely incorrect, these observations nonetheless provide valuable insights.

While the principle of neural networks draws inspiration from neurons, and certain architectures such as CNNs or transformers exhibit similarities, it's evident that real neurons and their interactions differ significantly from artificial neural networks by an order of magnitude. The internal complexity of real neurons is vastly different from their artificial counterparts. As a non-exhaustive list of differences, they can maintain distinct voltages across various components, accommodate diverse currents, and perform complex nonlinear operations. However, just as it would be inaccurate to attribute the tremendous success of deep learning solely to its parallels with the human brain, it is equally incorrect to dismiss the importance of the fact that these artificial neural networks originated from an attempt to mimic the functioning of the brain. While the differences are substantial, the foundational connection to biological neural processes underscores the significance of this inspiration.

3 Misleading intuitions from classical statistics

3.1 The complexity measures and generalization of state-of-the-art models

When talking about generalization, the corresponding theoretical framework in classic statistical learning is complexity measures. Such measures are supposed to quantify the effective capacity of a model family by evaluating its ability to fit the labeling of a data set in classification tasks. Different approaches exist, from combinatorial (VC complexity) to data-dependent (empirical Rademacher complexity) concepts, but for all of them, a high complexity is associated with poor generalization properties. The intuition behind this statement relies on the bias-variance trade off: if a model can perfectly fit a (training) data set, then it is very likely to not generalize well.

data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

Figure 1: From [16].The top-1 and top-5 accuracy (in percentage) of the Inception v3 model on the ImageNet dataset. Training and test accuracy are compared with various regularization turned on and off, for both true labels and random labels. The original reported top-5 accuracy of the Alexnet on ILSVRC 2012 is also listed for reference. The numbers in parentheses are the best test accuracy during training, as a reference for potential performance gain of early stopping

Though, in the case of state-of-the-art CNN, it has been shown empirically in [16] that vision models can achieve almost 100% training accuracy with both random labels (leading to a very high generalization error as there is no correlation between labels and inputs) and the true labels (with a very satisfying generalization error). This observation definitively challenges the aforementioned complexity measures as they tend to be empirically maximized in the above experiment, therefore not accounting for the good generalization properties. Indeed, as shown on Table.1, an Inception v3 architecture trained in the exact same framework, including regularizers or not, can achieve good generalization error while perfectly matching random labeling when trained on it. Table.1 also shows that explicit regularization methods such as weight decay or data augmentation still really improves generalization, even if it is not compulsory for a satisfying result. This improvement cannot be explained by the complexity concepts. Therefore one can draw two conclusions from this set of experiments:

1. Current theoretical complexity measures are not suitable for evaluating the performances of state-of-the-art vision models. New notions should better include the nature of the data and labels, as it naturally influences the generalization.

2. Explicit regularizers improve generalization, making the model somehow "simpler", but are not compulsory for already good generalization

Another quite astonishing result coming from [16] is illustrated on Fig.2 When varying the proportion of randomized labels in the data set, Inception v3 model always achieves a perfect matching on the training set and the test error varies similarly to the randomized proportion. *Thus, when the proportion of random label is intermediate, the model is able to extract meaningful information that applies for the test set, while still perfectly matching the training set.* Authors of [16] do not explain this behavior, as it illustrates the "weird" generalization properties of state-of-the-art CNN. A key question seems to be: how does the model deal with the information encoded in the training set, as it is partly randomized, partly meaningful? Brute-forcing the matching of the training set is it enough to extract the relevant information it contains that can apply to the test set?

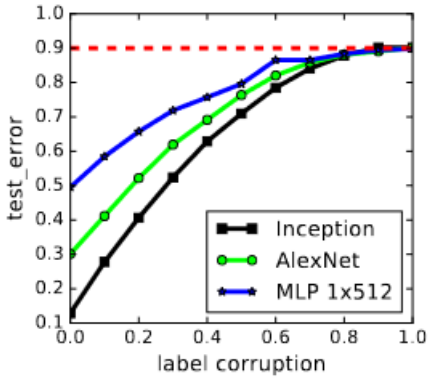


Figure 2: From [16]. Evolution of the test error (and generalization error, as training error is null) when training Deep CNN on CIFAR10 with a varying proportion of random labels.

A possible answer to this question is to consider that the learning process of a deep neural network is split in two different stages: memorizing the training set and producing a general understanding of both the data and targeted task. These stages do not necessarily happen successively. The previous hypothesis is quite consistent with the observations on grokking (see 7) and the representation of the data manifold (see 4.1). Let consider the case where 50% of the labels are random. Even if in this case the data manifold is hugely noisy, one could think that the model learns some features of the data manifold and is able to correctly interpret them. The representation (the direction) of these features in the learnt latent space will be erroneous, but this might be enough to correctly project some of the test images whose only features appear to be those identified by the model.

Just like most of the hypothesis given in the present document, none of these explanation is entirely satisfying but some empirical observations (see 4) guide us towards them.

3.2 Deep Double Descent

In the realm of machine learning, and as seen in the previsous subsection, the classical bias-variance trade-off has long guided our understanding of model performance. However, a perplexing anomaly, known as *Deep Double descent*, challenges it.

Deep double descent introduces a paradoxical scenario where, contrary to conventional wisdom, model generalization does not follow a U-shaped curve as a function of model size. Initially, as the number of parameters of the model grows, the test error follows the expected U trajectory, increasing with it. However, rather counter-intuitively, beyond a certain threshold of size, the test error starts decreasing again. Here again, this unexpected phenomenon challenges our fundamental assumptions about the relationship between model complexity and generalization error

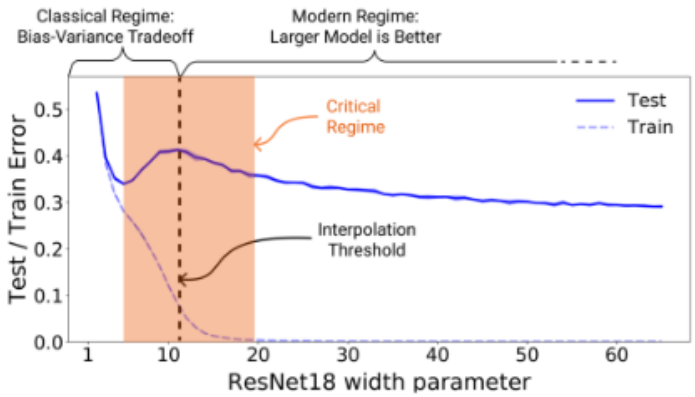


Figure 3: From [8], Double Descent phenomenon when looking at the train and test error as a function of model size, for ResNet18s of varying width on CIFAR-10 with 15% label noise

It introduces non-monotonic zones where increasing data does not consistently boost model performance, challenging our expectations and understanding. This disrupts this narrative by suggesting that test error can decrease after an initial rise, even as model complexity grows.

The intriguing aspect lies in the absence of a clear theoretical explanation for these non-monotonic zones. Unraveling this mystery is not just an intellectual pursuit but a crucial step in advancing our understanding of deep neural networks.

Works from the team at Anthropic [5] suggest that, in the small data set regime, models tend to memorize the training data, and to learn a more optimal representations for larger data sets, but, that the regime between the two is more mysterious. The model is no longer able to memorize the training data but is not building suitable representation.

The implications question even basic practical considerations in model development, where the conventional wisdom of "more data is always better" and "more complex models lead to overfitting" does not hold true in all cases. Deep double descent opens a new frontier in machine learning, challenging established norms. So, deep double descent is not merely a theoretical abstraction but a tangible challenge that machine learning practitioners confront in the quest for optimal model performance. While it appears that models with more parameters are able to leverage them to build better representations, the detail of *how* it happens remains elusive.

4 What may explain these observed behaviours?

In the following section we try to look under the hood of neural networks. We provide two main pieces of explanations regarding the performances of the models. We first focus on the internal data representation (4.1) before diving into interpretability-related topics (4.2.1).

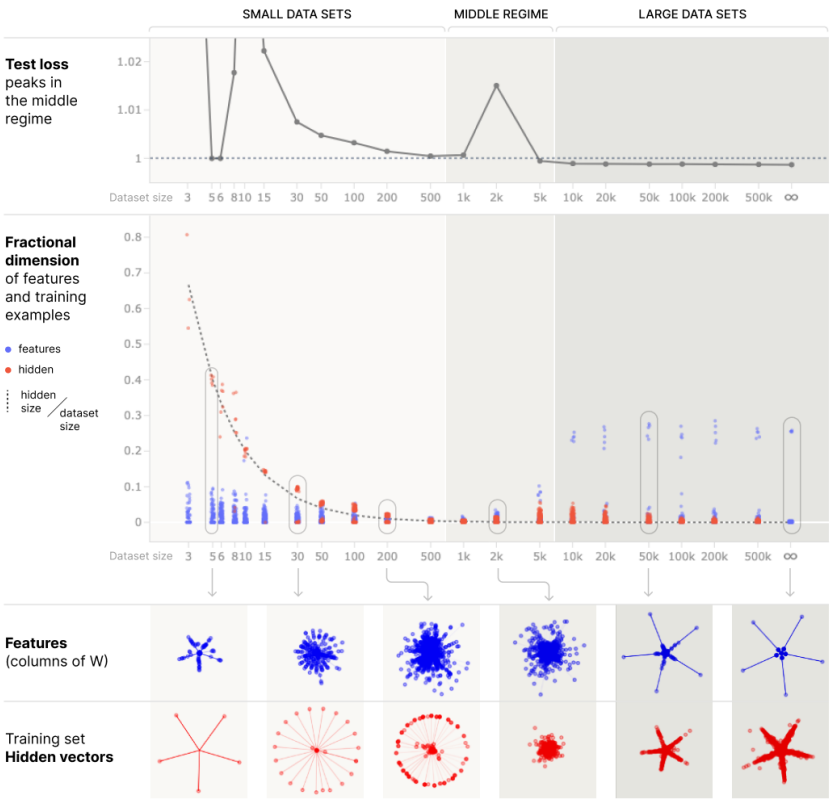


Figure 4: Evolution of learned features and training vectors embedding of a synthetic data set. From [5]. We do note detail the dataset or precise arguments here, but the "five pointed star" is an optimal way to represent it in two dimensions. At first, the model stores the n first training vectors in an n-pointed star. This eventually breaks down, n increases and surpasses the number of available parameters, but after significantly increasing

4.1 The paramount importance of data: exploring the manifold hypothesis and scaling law

4.1.1 The manifold hypothesis

The manifold hypothesis serves as a fundamental guiding principle in the realm of neural networks. This widely embraced concept posits that when dealing with high-dimensional data relevant to a specific task or reflective of a shared reality, these data points essentially reside on lower-dimensional manifold.

An established hypothesis, qualitatively validated in [2], suggests that when training a neural network, the model learns to map the data to a lower-dimensional manifold, thereby defining features relevant to the task. In stark contrast, traditional machine learning methodologies more heavily rely on the laborious manual construction of features, a process fraught with the potential difficulty of identifying relevant ones. Following feature extraction, linear or kernel methods are typically applied. Although some might argue for the use of advanced techniques such as Principal Component Analysis (PCA) to analytically define features, it's crucial to note that while PCA reduces dimensionality, it may not necessarily uncover the underlying manifold (in particular, since it learns the singular components of the covariance matrix of the distribution, it assumes that the manifold is an ellipsoid). Additionally, classical machine learning often employs kernel methods to address nonlinearities. However, these methods lack the adaptability to the data inherent in neural networks. They offer a fixed, non-data-adaptive approach. In essence, the manifold hypothesis underscores the paradigm shift brought about by neural networks, where the model autonomously learns and refines features on a lower-dimensional manifold, contrasting with the manual feature engineering and less adaptive nature of classical machine learning techniques.

4.1.2 Scaling law

In [13], it is emphasized that this underlying manifold depends on both the data and the task at hand. Furthermore, it is argued that scaling the size of the training dataset can be viewed as interpolation between points on the data manifold, gradually approaching the 'truth' manifold that would be defined if we had god-like access to the 'truth' infinite dataset. From this perspective, one might intuit that for a fixed given task, increasing the size of the dataset would enhance the performance of a given architecture. This idea is seemingly supported by the language model scaling law uncovered in chinchilla's paper [11] :

$$L(N, D) = \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}} + E$$

where N, D are respectively the model size and dataset size, and L the loss function. Indeed, empirical results showed that α is greater than β: the current bottleneck in language modeling performance lies in the constraint imposed by data size rather than the model itself (see Figure 5).

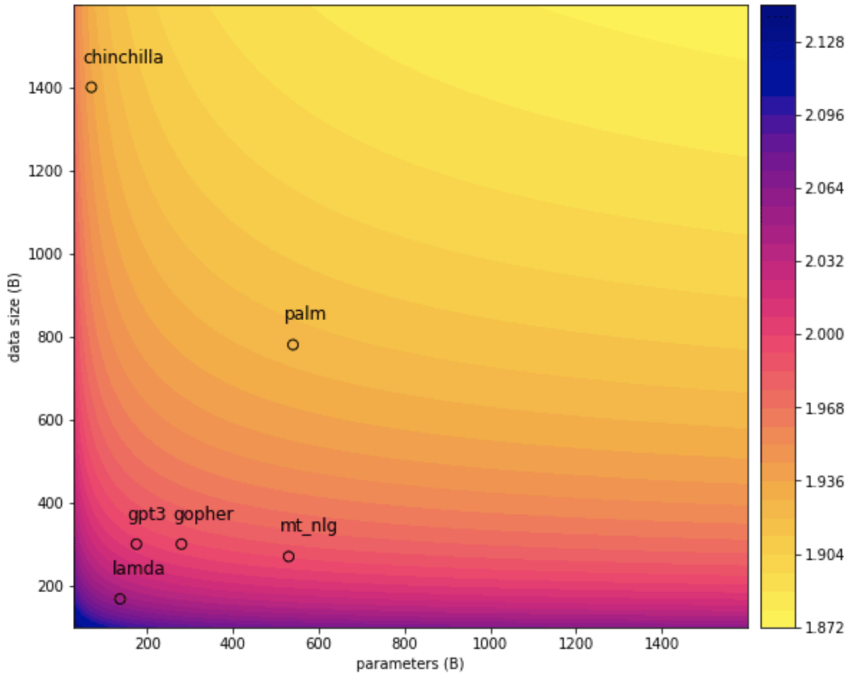


Figure 5: Language model loss: a function of data size and model size, from Chinchilla's wild implications.

This idea is further supported by comparing the representations learned by different CNNs. Interpretability research [10] has revealed that the representations learned by these models exhibit similar features (see curve detector features in Figure 6). In other words, the learned mapping to the underlying manifold appears to depend solely on the data rather than the architecture of the model, affirming the paramount importance of the data in influencing model performance.

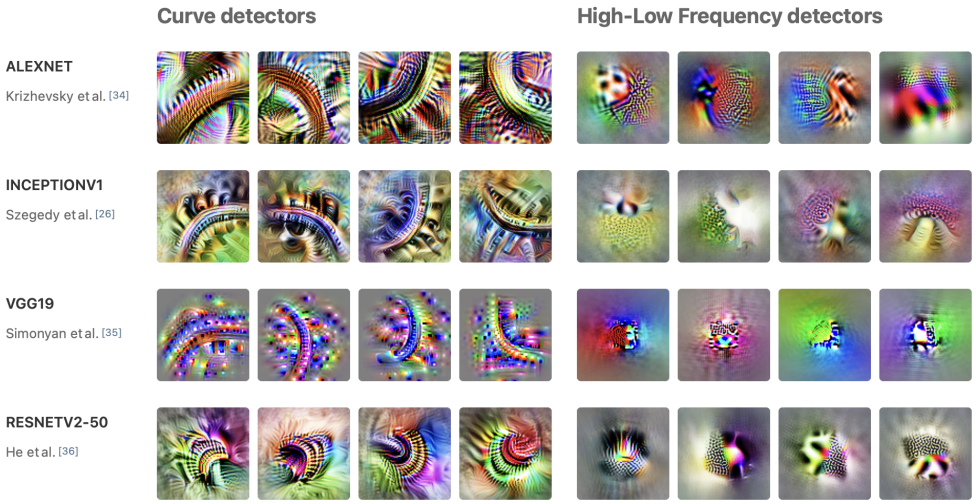


Figure 6: Curve detectors feature visualization for different CNNs.

4.2 Circuits in neural networks

4.2.1 Grokking

Grokking is one of the patterns that are very specific to deep learning. Its stydt might lead to good explanations of the performances and generalization properties of neural networks. In a few words, for some simple toy datasets and tasks like modular addition or similar binary operations, after first obtaining near 100% train accuracy but poor performance on the test set, the test loss suddenly drops after long enough training (this is usually done with small transformer decoder models), see Figure 7.

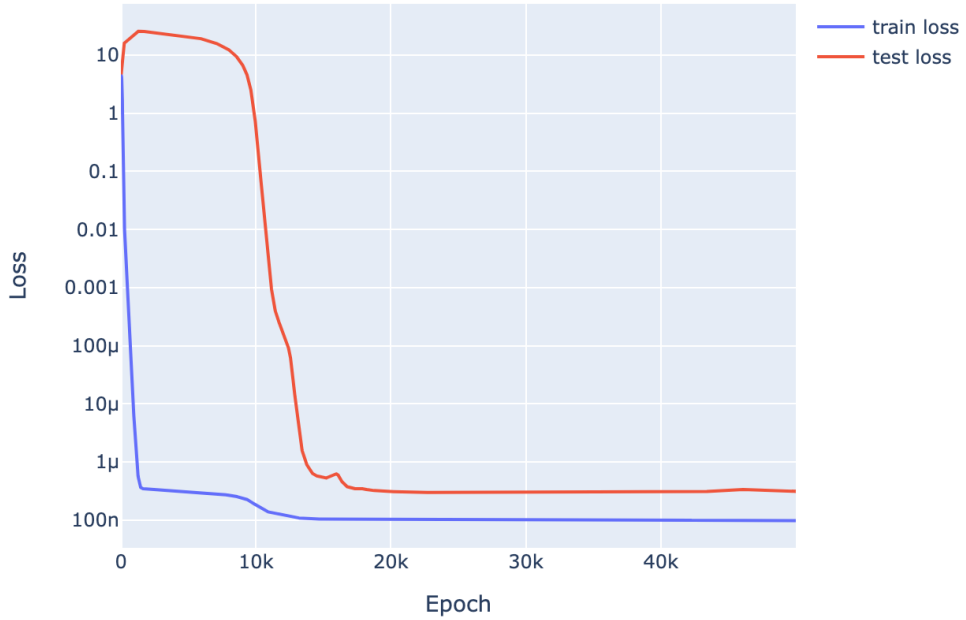
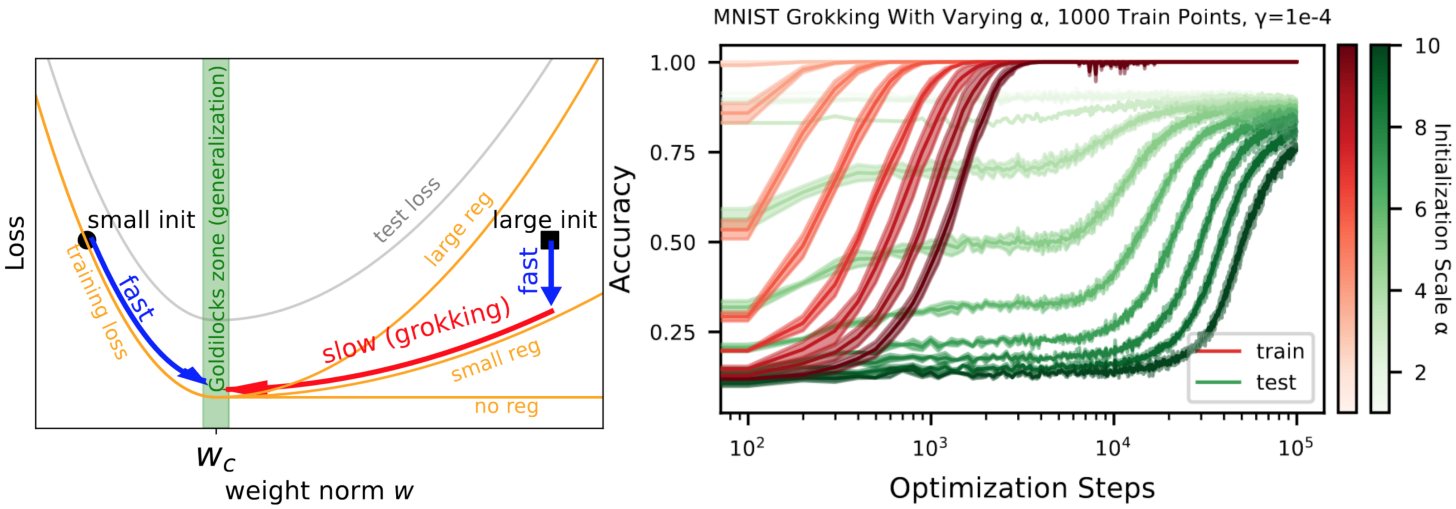


Figure 7: Grokking for modular addition ($P = 113$), from [9].

The circuits learned by a model which grokked modular addition have been fully reversed-engineered [9]. The most natural way to compute modular addition $\bmod P$ is by composition of rotations of angle $P/2\pi$ on the circle. The two numbers to be added are one hot encoded, and are mapped to $\sin(w_k a), \cos(w_k a), \sin(w_k b), \cos(w_k b)$ for $w_k = 2k\pi/P, k \in \mathbb{N}$ a frequency. Then $\cos(w_k(a+b))$ and $\sin(w_k(a+b))$ are computed in the attention and MLP layers, and used in the unembedding layer to compute $\cos(w_k(a+b-c))$ for each output, and the different frequencies interfere constructively at

$c^* = a + b \bmod P$ and destructively elsewhere, giving high probabilities for c^* and low for all other c. What happens over the training process can be split up in 3 stages:



(a) Neural networks loss as a function of weight norm

(b) Grokking MNIST

Figure 8: Importance of weight norm for generalisation, from [7].

1. Memorization: the model learns all the train data and train loss goes down to zero
2. Circuit formation: over time, the model starts to learn better representation (forced by the weight decay, see next paragraphs)
3. Grokking: the circuit is complete, the test loss drops abruptly near zero

While grokking was first observed in simple, algorithmic datasets, there is evidence that it can happen in tasks such as image classification. Authors of the Omnigrok paper [7] argue that grokking takes place in contexts where the representation is critical. In the case of MNIST, the original representation is already pretty good: you can have 95% train accuracy with a dumb k-nearest-neighbours classifier, eventually climbing up to 99% with smarter choices of hyper-parameter. This explains why, for MNIST (given a reasonable architecture), the test loss follows the training one. For modular addition, the original representation (1-hot encoded vectors) is catastrophic: when we start training the model, the train accuracy jumps up, close to 100%, while the test accuracy stays (at first) stays close to zero. It catches-up with the train accuracy only when the model grokks the right representation of the problem.

A general observation has been that, the lower the norm of the weights of the network, the better the representation. If you initialize the network with large weights, even for a problem with a good initial representation, the network will at first be forced to memorize the bad representation, before learning better ones over time, see 8(a).

From these observations, it becomes possible to force a model to grokk on data which has already good representations by initializing the model with large weights, see 8(b) for an example with MNIST.

4.2.2 Induction heads

First described in [11], induction heads appear to be the main driver of the ability of transformer language models to leverage context information (at least in the simpler models, where the number of layers stays small). Induction heads attend to the previous occurrence of the current token, and predict the next token according to the subsequent token of the previous appearance, see Figure 9.

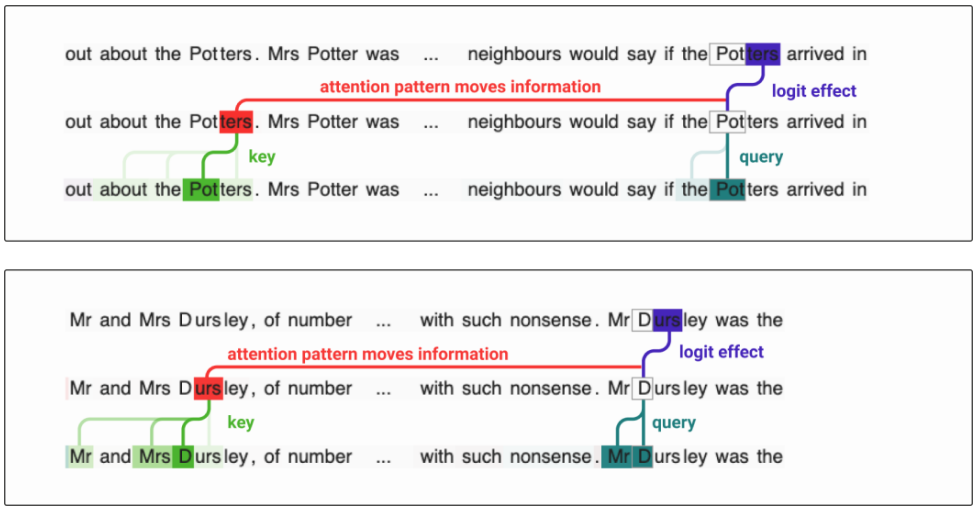


Figure 9: Induction heads mechanism, from [11].

Their formation during the training stage results in a significant drop in the loss. It is associated with several phase changes in various statistics, the prefix matching score, see Figure 10. This figure also illustrates that induction heads are only formed in models with at least two attention layers. We can also remark that heads that become attention heads go suddenly, from not doing prefix matching at all to clearly doing it. This shows behavior strikingly similar to that of grokking.

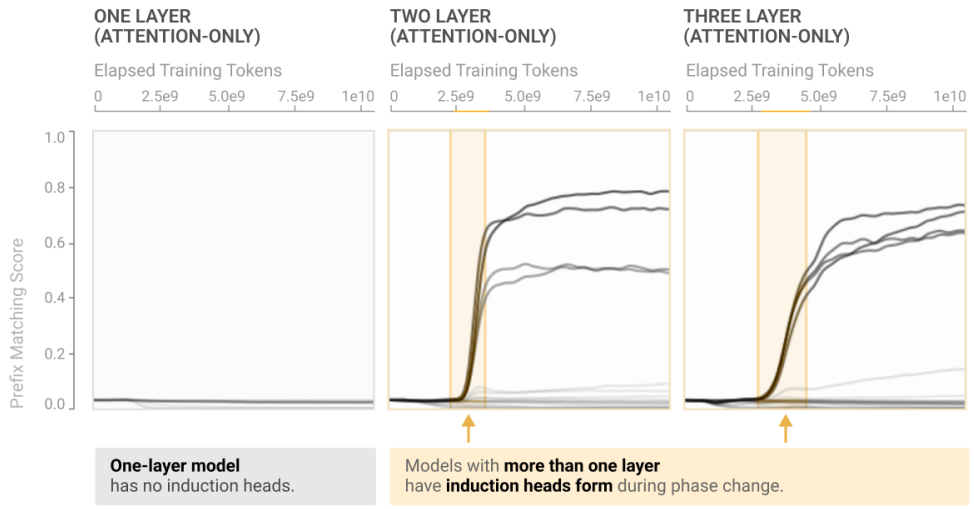


Figure 10: Induction head formation.

5 Insights into neural networks learning

In the last section, we give additional details that may help to better understand neural networks behaviours, even if not directly explaining the generalization properties we talked about before.

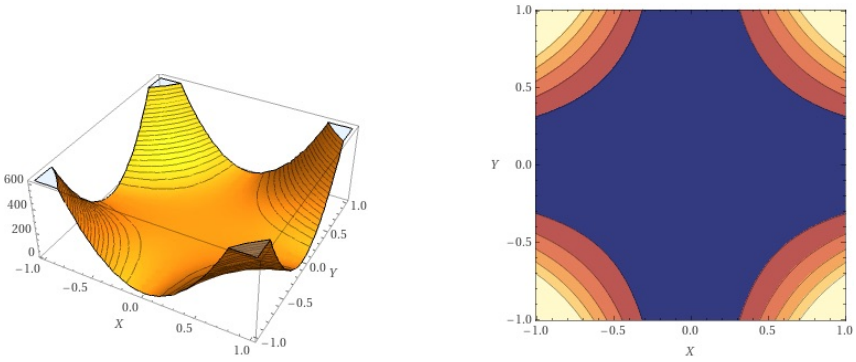
5.1 Singular Learning Theory

In deep neural network, we could find all lot of matrix products between weights and inputs. These matrix products offer a lot of symmetries [6]:

These symmetries are interesting but common as we did not assume anything on the weights. They are known as generic symmetries. To go further, we can study non-generic symmetries that are linked to the structures of the weights. These symmetries authorize us to permute weights and, by doing so, to

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \cdot \begin{pmatrix} j & k & l \\ m & n & o \\ p & q & r \end{pmatrix} = \begin{pmatrix} b & a & c \\ e & d & f \\ h & g & i \end{pmatrix} \cdot \begin{pmatrix} m & n & o \\ j & k & l \\ p & q & r \end{pmatrix}$$

access not to point of minimum loss but surfaces. These surfaces of minimal loss could have the following shape in an toy model. The followings figures (11) are really illustrative of that phenomenon.



(a) 3D plot of a loss function according to two weights

(b) Level set of a loss function according to two weights

Figure 11: Loss function.

We find in those graphs, a "cross" which could be explained by the presence of a singularity.

The uniform loss values across every point on this minimal loss surface raise questions about the nature of these surfaces and the underlying factors contributing to such homogeneity.

Going deeper into neural network optimization, this uniformity implies a multitude of weight configurations resulting in the same loss. Yet, discerning an optimal point within this ensemble becomes a pivotal task, introducing the need for advanced optimization strategies.

In this context, the adoption of a Bayesian optimization approach appears as a powerful tool. Bayesian optimization allows for the intelligent exploration of the weight space, considering not only the loss values but also the uncertainty associated with them. By incorporating probabilistic models and iteratively selecting points to evaluate, Bayesian optimization navigates in the neural network landscape with a nuanced perspective. This perspective enables the discovery of optimal points that align with broader objectives, such as improved generalization, robustness, or computational efficiency. This adaptive exploration of the weight space, guided by probabilistic reasoning, introduces a level of sophistication in optimization strategies, especially crucial in the framework of high-dimensional and complex neural network architectures.

This multifaceted understanding not only enhances our ability to train more efficient and effective neural networks but also paves the way for innovative advancements in the field of deep learning. However, the Bayesian optimization is still a theory and a field of research and has not yet been solved.

5.2 Models interpolate or extrapolate?

A simple question one can try to answer when thinking about the performance of a machine learning model is: is it good at interpolating? extrapolating? Both of them? Let first correctly introduce these concepts, using the definitions from [1]:

Interpolation (extrapolation) occurs for a sample x whenever this sample does (not) belong to the convex hull of a set of samples $X = \{x_1, \ldots, x_N\}$.

Answering this question is a good way to better understand how a model works, and should provide a more realistic visualization of its capabilities. As explained more in details in 4.1 one may think that generalization properties rely on the following process:

1. learning a good representation of the data manifold

2. correctly interpolating the targeted function on this latent representation

Authors of [1] empirically showed that for the probability of the event {*a new data point belongs to the convex hull of the training set*} to be close to 1, the size of the training set grows exponentially with the dimension of the manifold. This statement is consistent with theoretical results presented in (see [3]). Therefore, it appears that even in the embedding space, machine learning models do not interpolate. This is quite intuitive for classification tasks, when one expects the model to shatter the input data in a (learned) representation space to better split the different labels. Though, for more complex and continuous tasks, one should not stick to the common representation associated with (linear) regression for extrapolation. Somehow, modern architectures are able to produce very good extrapolated answers, accounting for their good generalization properties. Thus, contrary to most classic machine learning methods, increasing the size of an architecture with suitable regularization processes does not amount to complexifying and refining the approximation of the targeted function on the complex hull of the data manifold. These properties can be linked with circuit learning in transformers (cf observations on grokking 4.2.1) and representation learning in Deep CNN. For instance, a possible explanation for the generalization properties of such models would be to think of the representation of an input image as the linear combination of features learnt in the embedding space, which does not depend on the convex hull of the training set.

6 Conclusion

The current rapid advancements in deep learning owe much of their success to the interdisciplinary nature of contemporary research, a landscape where diverse insights from fields such as computer science, neuroscience, physics, and mathematics converge. This convergence has become a key catalyst, propelling innovations at an unprecedented pace. Though, some of the recently achieved results remain mysterious. To foster a more reliable and aligned AI, a better understanding of the deep learning is required, as concepts and tools from the classical statistics theory are not enough.

In this document, we tried to cover some of the key observations proving that the neural networks do not behave as expected. Their surprisingly good properties should guide us towards the deeper understanding of these models. We explore a few of the underlying hypothesis but we must acknowledge that none of them are entirely satisfying.

References

[1] Randall Balestriero, Jerome Pesenti, and Yann LeCun. Learning in high dimension always amounts to extrapolation. *CoRR*, abs/2110.09485, 2021.

[2] Pratik Prabhanjan Brahma, Dapeng Wu, and Yiyuan She. Why deep learning works: A manifold disentanglement perspective. *IEEE Transactions on Neural Networks and Learning Systems*, 27(10):1997–2008, 2016.

[3] Imre Bárány and Zoltán Füredi. On the shape of the convex hull of random points. *Probability Theory and Related Fields*, 77, 1988.

[4] Radoslaw Martin Cichy, Aditya Khosla, Dimitrios Pantazis, Antonio Torralba, and Aude Oliva. Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific Reports*, 6(1):27755, Jun 2016.

[5] Tom Henighan, Shan Carter, Tristan Hume, Nelson Elhage, Robert Lasenby, Stanislav Fort, Nicholas Schiefer, and Christopher Olah. Superposition, memorization, and double descent. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/toy-double-descent.

[6] Jesse Hoogland. Neural networks generalize because of this one weird trick. `https://www.lesswrong.com/posts/fovfuFdpuEwQzJu2w/neural-networks-generalize-because-of-this-one-weird-trick`, 2023.

[7] Ziming Liu, Eric J Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*, 2023.

[8] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *CoRR*, abs/1912.02292, 2019.

[9] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023.

[10] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization.

[11] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[12] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[13] Utkarsh Sharma and Jared Kaplan. Scaling laws from the data manifold dimension. *Journal of Machine Learning Research*, 23(9):1–34, 2022.

[14] James C. R. Whittington, Joseph Warren, and Timothy Edward John Behrens. Relating transformers to models and neural representations of the hippocampal formation. *CoRR*, abs/2112.04035, 2021.

[15] Yaoda Xu and Maryam Vaziri-Pashkam. Limits to visual representational correspondence between convolutional neural networks and the human brain. *Nature Communications*, 12(1):2065, Apr 2021.

[16] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016.