Evaluation of Subspace Training and Intrinsic Dimension, extending with FastJL Projection

January 9, 2023

Antonio Andrea Gargiulo

Abstract

Works like lottery tickets, distillation, pruning and training in subspaces demonstrate that neural networks are mostly **over-parametrized**. It would be possible to train them with fewer parameters or much smaller degree of freedom. In this work, we studied the main property of this variety of techniques. We have performed extensive experimental research on the **subspace training** methodology to understand the benefits and drawbacks of this training technique. Moreover, we have developed a different type of projection: **FastJL** and compared its performance with the other projection methods.

1. Introduction

What happens when, instead of optimizing an entire neural network's parameters' space, we **project** the parameters and allow the optimizer to move only in a lower dimensional subspace? Is a random subspace good enough to still find a reliable solution? It depends on the **dimension** of the subspace chosen, and this surprise no one because we can choose from one dimension to the entire dimensions of the space as our subspace. But when does the solution start to appear? If we call this point: **intrinsic dimension**, what does it tell us about the properties: of the network, of the solution and about the data? In this report, we tried to find an answer to those questions, studying and researching the main property of this family of learning models.

2. Related work

The **landscape** of the optimization problem is **simpler** than we think, as stated in Goodfellow et al. (2014), and this can make us assume that the intrinsic dimension could be only an upper bound of the real dimension of the problem. Over time, as our neural networks and datasets became large and more complex, new information emerged, and studies were made, such as Frankle (2020) and a measure as the intrinsic dimension became more relevant to characterize the space in which the optimization problem takes place and to provide a more detailed evaluation to **compare** models and their sub-models (Li et al., 2018). Recently, Larsen et al. (2021) showed a theoretical explanation of the phase transition that brings to the intrinsic dimension with a geometrical interpretation. They also showed where and why this transition happens and its dependence on the quality of the **initialization**. As a result of this recent work, we can interpret those results as in Figure 1. As depicted, when we limit the dimensionality of the network, projecting the parameters in a lower dimensional space, it is still possible for the optimization process to reach a reasonable solution with fewer redundant parameters if the subspace is **oriented towards** the true and the projected solutions.

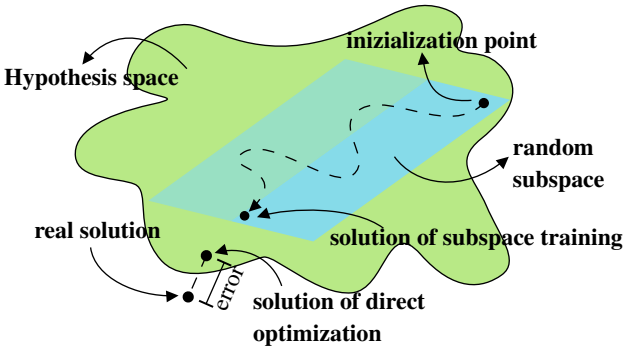

Figure 1. The hypothesis space is given by the construction of the network's architecture and the dataset (it is high dimensional, but for obvious reasons is pictured as 3-dimensional). The real solution can instead be interpreted as a point that may lie inside or outside the fixed hypothesis space, and its projection on this space is the solution of the direct optimization procedure. The magnitude of the projection itself can be interpreted as the error (in a noise-free setting). The subspace training procedure can reach a solution spanned by the selected random subspace (here pictured as a plane).

Email: Antonio Andrea Gargiulo <gargiulo.1769185@studenti.uniroma1.it>.

Deep Learning and Applied AI 2022, Sapienza University of Rome, 2nd semester a.y. 2021/2022.

3. Method

The subspace training of a model with a D-dimensional parameters space in a d-dimensional subspace(with$D >> d$) can be intuited through the following formula, expressing the **subspace iterative minimization procedure**:

$$\theta_{t+1}^D = \theta_t^D - \alpha \nabla \ell_\theta(S(\theta_t^D)) \tag{1}$$

where S is the **projection operator** that maps the full parameters space in the subspace, defined as:

$$S(\theta_t^D) = \theta_0^D + P\theta_t^d \qquad P \in \mathbb{R}^{D \times d} \tag{2}$$

Instead of optimizing the whole θ^D, we optimize only θ^d. From the eq. 2 we can clearly see that the quality of initialization θ_0 can directly affect the algorithm's performance, as already referred in Larsen et al. (2021), and also because it remains present in each step of the optimization. Like in a **pre-trained** architecture where we find a good initialization point and then **fine-tune** it on new data, which we expect to live in a space **similar** to the ones used for pre-training. "This effect is akin to staring out into the night sky in a single random direction and asking with what probability we will see the moon; this probability increases the closer we are to the moon." - Larsen et al. (2021).

We tested the subspace training on several **architectures**: Fully Connected, LeNet, various LeNet variants (with tied-weights, local connections and maximal convolutions) and ResNet-18; different **datasets**: MNIST and CIFAR-10 and with different **projection algorithms**: dense, sparse, Fastfood and FastJL. The different projection algorithms are necessary to allow some architectures to reach the intrinsic dimension because the dense projection is very eager in memory consumption, so we need to take a better approach to compute the projection matrix needed in the eq. 2.

The **FastJL projection** (Ailon & Chazelle, 2006; Fandina et al., 2022; Thickstun, 2016) is a fast and memory efficient method to compute a sparse projection, based on the **Johnson-Lindenstrauss lemma** (Johnson et al., 1986; Larsen & Nelson, 2016). Basically, this transformation efficiently multiplies 3 matrices PHD, where P is valued with a Bernoulli distribution, with probability $q = 2ln^2(N)/d$, of 0 and values from a Normal distribution of variance $q^{-1/3}$ and 0 mean; H is a FW-Hadamard matrix and D is a random reflection Rademacher diagonal matrix.

4. Experimental Results

In our study, we were able to reproduce the results of Li et al. (2018) on MNIST and CIFAR-10, with mostly similar measurements on the intrinsic dimension. In the majority of the cases, our implementation reaches the intrinsic dimension **slightly before** the original paper, and this brings us to the conclusion that this value is only an upper bound (an estimate) of a more deep phenomenon that can capture **an important characteristic** of the optimization landscape. We show a summary of the results in Table 1, 2 and a more comprehensive one on: github.com/AntoAndGar/Intrinsic-Dimension. As we can see from the measured intrinsic dimension, the results are reliable, as the main paper, also us have found that the MNIST problem is **easy** than CIFAR-10 and the **CNN** architectures (LeNets) are better at classifying images than Fully Connected ones. Instead, our results on the various LeNet variants show more confusion about which are the **essential properties** of a CNN. Since the intrinsic dimension is much more of an estimate than Li et al. (2018) claim, we do not think this feature can be measured using only this comparison parameter.

The FastJL projection shows better performance in terms of **speed** w.r.t. the Fastfood projection, and the obtained results (see Table 3) are almost similar to the Fastfood transform. The only drawback of this transformation is when the d_{int90} grows a lot. Because, by the property of the FastJL, it decreases the density of the projection matrix and increases the variance of the Normal distribution very aggressively. In addition to being **highly sparse** and with **high values** in the few parameters, networks with this structure become difficult to train and highly unstable (without regularization techniques).

5. Conclusions

The subspace training method shows several important properties of the optimization landscape, including the ability to **compress** the obtained model and determine which type of architecture is **best suited** for a particular problem. Performances of the final models are not as good as those of standard training models. In addition, this method does not apply to devices that can benefit from the small and compressed model since the **resources** required for putting the model into production are nearly equal to those for standard networks. It, therefore, limits the advantage of having compressed models only to cases where they must be **transmitted over a network**. Another argument against this method is that we can achieve the same validation accuracy with **smaller networks** of comparable size of the intrinsic dimension if we tune the hyperparameters well on those problems without the need for the complexity of subspace training. The intrinsic dimension, instead, is a useful metric to understand the **complexity** of the optimization landscape, however, it is not as reliable as we would like it to be because we have found that it is only an **estimate**, and we think that it should be more robust to the change of hyperparameter or it must follow the common best practices for the training of neural networks and not only the ones proposed in Li et al. (2018).

MNIST		
Network Type	**Parameter Dim. D**	**Subspace Dim. d**
FC	199,210	600
LeNet	44,426	170
Untied LeNet	286,334	350
FC LeNet	3,640,574	900
FC Tied LeNet	193,370	400

Table 1. Measured d_{int90} on MNIST dataset for the various architectures, the value of intrinsic dimension, in this case, is measured when the model reach 90% of validation accuracy.

CIFAR-10		
Network Type	**Parameter Dim. D**	**Subspace Dim. d**
FC	1,051,930	5,000
LeNet	62,006	600
Untied LeNet	658,238	10,000
FC LeNet	16,397,726	5,000
FC Tied LeNet	297,734	3,000
ResNet	292,954	1,000

Table 2. Measured d_{int90} on CIFAR-10 dataset for the various architectures, the value of intrinsic dimension, in this case, is measured when the model reach 45% of validation accuracy and 52.2% for ResNet (as in the main paper).

CIFAR-10 - FastJL Projection		
Network Type	**Parameter Dim. D**	**Subspace Dim. d**
FC	1,051,930	4,000
LeNet	62,006	1,250
Untied LeNet	658,238	2,000
FC LeNet	16,397,726	5,000
FC Tied LeNet	297,734	2,500
ResNet	292,954	1,000

Table 3. Measured d_{int90} on CIFAR-10 dataset for the various architectures with the FastJL projection method, the value of intrinsic dimension, in this case, is measured when the model reach 45% of validation accuracy and 52.2% for ResNet.

References

Ailon, N. and Chazelle, B. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pp. 557–563, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595931341. doi: 10.1145/1132516.1132597. URL https://doi.org/10.1145/1132516.1132597.

Fandina, O. N., Høgsgaard, M. M., and Larsen, K. G. The fast johnson-lindenstrauss transform is even faster, 2022. URL https://arxiv.org/abs/2204.01800.

Frankle, J. Revisiting "qualitatively characterizing neural network optimization problems", 2020. URL https://arxiv.org/abs/2012.06898.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. Qualitatively characterizing neural network optimization problems, 2014. URL https://arxiv.org/abs/1412.6544.

Johnson, W. B., Lindenstrauss, J., and Schechtman, G. Extensions of lipschitz maps into banach spaces. *Israel Journal of Mathematics*, 54:129–138, 1986.

Larsen, B. W., Fort, S., Becker, N., and Ganguli, S. How many degrees of freedom do we need to train deep networks: a loss landscape perspective, 2021. URL https://arxiv.org/abs/2107.05802.

Larsen, K. G. and Nelson, J. Optimality of the johnson-lindenstrauss lemma, 2016. URL https://arxiv.org/abs/1609.02094.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes, 2018. URL https://arxiv.org/abs/1804.08838.

Thickstun, J. The fast johnson-lindenstrauss transform. 2016. URL https://johnthickstun.com/docs/fast_jlt.pdf.