Final Project: Foundation of Inference and Learning

Bohan Wang
bohan.wang@epfl.ch

Ke Wang
k.wang@epfl.ch

Xinghai Wang
xinghai.wang@epfl.ch

***Abstract*—Although deep neural networks often have far more parameters than the number of training samples, they exhibit a remarkably small difference between training and test performance, known as generalization error. The paper *Understanding Deep Learning Requires Rethinking Generalization* [1] challenges the conventional assumption that low generalization error is attributed to regularization techniques through both experiments and theoretical constructions. Their results suggest that the traditional understanding of regularization and generalization fails to explain the astonishing generalization capabilities of neural networks. In this report, we reproduce the results and critically discuss their conclusions through well-designed experiments.**

I. Introduction

Successful deep artificial neural networks can achieve a remarkably small generalization error even with massive parameter size, while some architectures generalize poorly. A convincing theory to distinguish and explain the generalization behavior of neural networks is of vital importance, since it would lead to better interpretability and more reliable architecture design of neural networks [1].

Previously, traditional learning theories propose complexity measures of controlling generalization error, and suggest that regularization techniques are required in training to ensure small generalization error [2], [3], [4]. Zhang et al. [1] question the traditional view of generalization, by showing that it is incapable of explaining why neural networks have radically different generalization performance, through several carefully designed experiments. In this report, we will reproduce their results, and critically develop and extend their discussions associated with verified experiments.

This report is organized as follows, we reproduce the results of effective capacity of neural networks in Sec. II, the role of explicit regularizers in Sec. III, and finally reproduce the results for implicit regularization in Sec. IV.

II. Effective Capacity of Neural Networks

To understand the effective capacity of neural networks, the authors propose to train the network on a dataset, where there is no relationship between instances and their labels. Surprisingly, the selected neural networks could still perfectly fit the data, without substantially slowing down the convergence.

A. Reproduce results

The destruction of relations between instances and labels can be done by either corrupting the labels or corrupting the images. The following modifications are made to the dataset:

- **True labels**: the original dataset without modification.
- **Partially corrupted labels**: independently with probability p, the label of each image is corrupted randomly.
- **Random labels**: all the labels are replaced randomly.
- **Shuffled pixels**: a random permutation of the pixels is chosen and then the same permutation is applied to all the images in both training and test set.
- **Random pixels**: a different random permutation is applied to each image independently.
- **Gaussian**: A Gaussian distribution (with matching mean and variance to the original image dataset) is used to generate random pixels for each image.

The above modifications are made on the CIFAR10 dataset [5], where each image has a dimension of 3x28x28 and has 10 distinct labels. The selected candidate structures for this test are Inception[6], AlexNet[7] and MLP 1x512 (MLP with a single hidden layer). Note that instead of the original Inception and AlexNet, a smaller version of Inception and AlexNet are used on CIFAR10, whose structures are detailed in Appendix. We will refer to them as Inception and AlexNet in the remaining report.

1) Experimental details:

Pre-processing The pixels values are divided by 255 after the images are cropped from the center to 28x28. Then the images are normalized by subtracting mean and dividing by standard deviation for each image independently. This was done using `per_image_whitening` of TensorFlow [8] in [1]. We write a function to mimic its behavior with PyTorch [9].

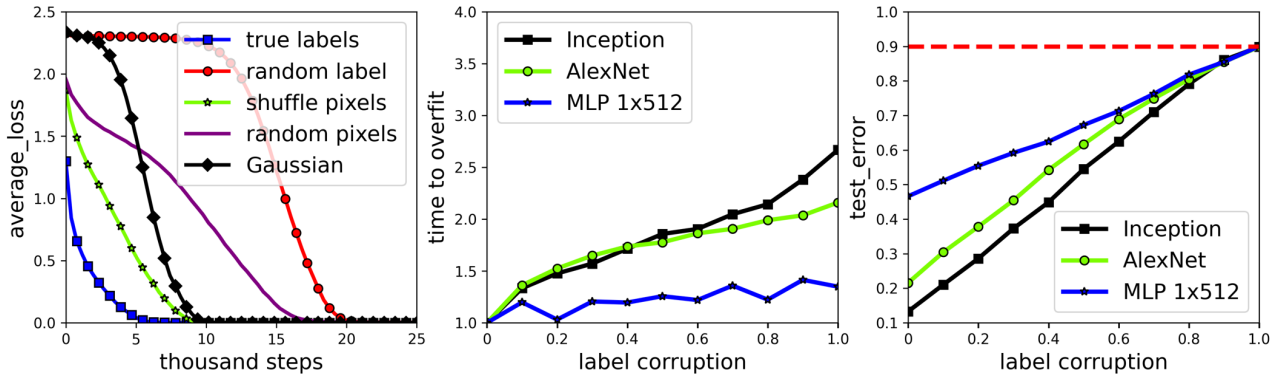
Training All experiments use SGD optimizer with a momentum of 0.9, using a batch size of 128. The learning rate is initialized to 0.1 for Inception and 0.01 for AlexNet and MLP 1x512, and decays in each epoch. In [1] the decay factor is 0.95 for all three models, but in practice we find a decay factor 0.98 is required for MLPs to converge to nearly 0 train error due to its relatively lower convergence speed.

2) Experimental results:

Fig. 1(a) presents the learning curves of Inception on CIFAR10 dataset, under various modifications on the data. Our results show the same behavior as the results in [1]: Inception can still fit the dataset perfectly with 0 train error, even though the relationship between images and labels is totally destroyed. Besides, several key observations can also be found on our result:

1) The training loss converges very fast once fitting starts.
2) Training with modifications on input images ("random pixels" and "Gaussian") converge faster than modifications on the labels ("random labels").

The behavior of three neural networks are examined with varying level of label corruptions. The time taken for overfitting the train set is illustrated in Fig. 1(b) for label corruption



(a) learning curves (b) convergence slowdown (c) generalization error growth

Fig. 1: Results for fitting CIFAR10 with randomm labels or pixels. (a) training loss curve of various settings. (b) relative convergence time with respect to varying label corruption level (c) generalization error (test error) with respect to varying label corruption level.

level from 0 to 100%. Our result draws the same conclusion that the convergence time only increases by a constant value with corrupted labels. Besides, the generalization error, which is in fact the test error since train error converges to 0, is shown in Fig. 1(c). As we can see, the generalization error grows with the label corruption level and finally converges to 90%, exactly the error by chance.

B. Discussions

1) Capacity of neural networks:

The results in [1] shows that the capacity of neural networks is large enough to memorize the whole CIFAR10 dataset of 50,000 images, with less than 2 million parameters. Although this is already quiet impressive, we further show that the neural networks actually possess higher capacity than what is shown in [1] with the following two experiments:

- **Fit Gaussian noise dataset 10 times larger:** From the authors' analysis on finite-sample expressivity [1], the two-layer MLP 1x512 should be able to perfectly fit the dataset, with a parameter number $p \geq 2n + d$. Given that the three models all have a total parameter over 1,000,000, they should be able to represent a function of a sample size 500,000. We create a dataset containing 500,000 images of Gaussian noise with random labels and fit the models. The results show all three models perfectly fit the train set with 0 error, verifying the authors' analysis.
- **Fit MNIST with random label:** With random labels, the hand-written digits in MNIST dataset [10] can be much harder to fit compared with CIFAR10, because the instances in MNIST with the same label are much more similar than the instances in CIFAR10. Our experiment shows that, despite many epochs required, the three aforementioned models are still able to perfectly fit the MNIST dataset with random labels, with over 99% accuracy. This result is a much stronger implication for their capacity, by showing that the capacity of the neural networks is able to memorize slight differences among instances.

2) Recognizing pattern VS. memorizing with brute-force:

The result of this section is very important, as it proposes the question: does neural networks learn by recognizing an intrinsic pattern (which is the general assumption), or it simply memorizes the entire train set using brute-force?

The authors conclude that, when both pattern and noise exist in the train set, the neural networks are able to capture remaining signal in the data, while at the same time fit the remaining noisy part using brute-force [1], by showing the generalization performance is only partially damaged when the labels are partially corrupted. This can also be understood as, as long as some pattern exists in the train set, the models can always prioritize learning the pattern in the training process.

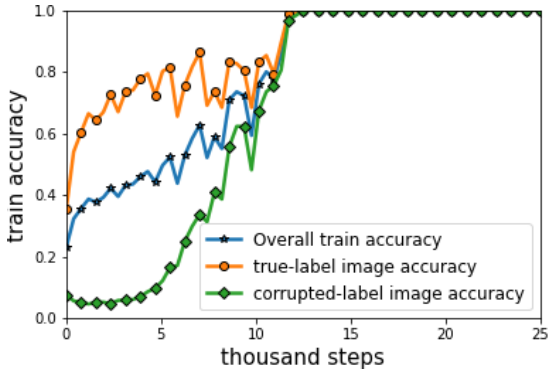


Fig. 2: Training accuracy curve for CIFAR10 with half labels corrupted.

Here we aim to confirm and visualize the priority of learning intrinsic pattern over memorizing noise in the training process, through the following experiment. We create a train set with half true labels (contains an intrinsic pattern) and half corrupted labels (represent noise). Therefore, the accuracy on the true-label images reflects the degree of learned patterns and the accuracy on the corrupted-label images reflects the degree of memorized noises. We visualize the learning curve, with the accuracy on true-label images and corrupted-label images

TABLE I: The training and test accuracy (in percentage) of various models on the CIFAR10 dataset

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.12
		yes	no	100.0	88.96
		no	yes	100.0	85.51
		no	no	100.0	85.27
(fitting random labels)		no	no	100.0	9.97
Inception w/o BatchNorm	1,649,402	no	yes	100.0	84.98
		no	no	100.0	84.03
(fitting random labels)		no	no	100.0	9.88
Alexnet	1,375,690	yes	yes	98.68	83.73
		yes	no	99.87	82.66
		no	yes	100.0	79.13
		no	no	100.0	78.99
(fitting random labels)		no	no	100.0	10.05
MLP 3*512	1,735,178	no	yes	99.98	55.69
		no	no	100.0	55.12
(fitting random labels)		no	no	100.0	10.51
MLP 1*512	1,209,866	no	yes	99.96	53.24
		no	no	99.97	52.62
(fitting random labels)		no	no	99.84	10.67

visualized separately, on Fig. 2. The curves in Fig. 2 clearly show that the models do learn the intrinsic patterns first in the training process, and then memorize the noises using the remaining capacity with brute-force in later epochs.

III. The Role Of Explicit Regularization

Explicit regularizers are traditionally considered as necessary tools to control generalization error. In this section the authors conclude that explicit forms of regularization, such as weight decay and data augmentation, do not adequately explain the generalization error of neural networks via the following experiments.

A. Reproduce results

The performances of Inception, AlexNet and MLP (with either 1 or 3 hidden layers) are compared with and without equipped explicit regularizers (including random crop and weight decay) on CIFAR10. Besides, the role of regularizers in fitting random labels is also studied.

Training details: The weight decay factor is set as 5e-5. Random cropping randomly crops the images from 3x32x32 to 3x28x28. The other pre-processing and training details are consistent with Sec. II-A1.

The results are shown in Table I, switching the use of random crop and weight decay, we can observe the following:

1) Explicit regularizer increases the test accuracy of models, while the train accuracy almost does not differ much, indicating explicit regularization indeed improves generalization performance.
2) Even with all of the regularizers turned off, all of the models still generalize well, suggesting lack of regularization does not necessarily introduce a dramatically larger generalization error.

B. Discussion

In this section, the authors question the role of explicit regularizers in generalization and come to the following conclusions:

1) Explicit regularizer is not a key in neural networks' generalization ability, but is more like a tuning parameter that often helps improve test error, since the models still generalize well even without explicit regularizers.
2) Explicit regularizer plays a rather different role in deep learning, in contrast to the general understanding of regularizers in classical convex minimization to confine learning to a subset of the hypothesis space with manageable complexity.

About the second conclusion, from classical view on explicit regularizers, the effective Rademacher complexity of the possible solutions would be dramatically reduced by adding an explicit regularizer, which would prevent them from overfitting a randomly-labeled dataset.

To confirm that the complexity of the neural networks are still large enough tho fit randomly-labeled dataset, we fit CIFAR10 with random labels using Inception, AlexNet and MLPs with explicit regularizer (weight decay) turned on. The results are given in Table II, where all models perfectly fit the dataset. The result suggests that the models still have very large complexity and is important since it requires people to rethink the role of explicit regularizers in generalization of deep learning, as opposed to their role in classical convex empirical risk minimization.

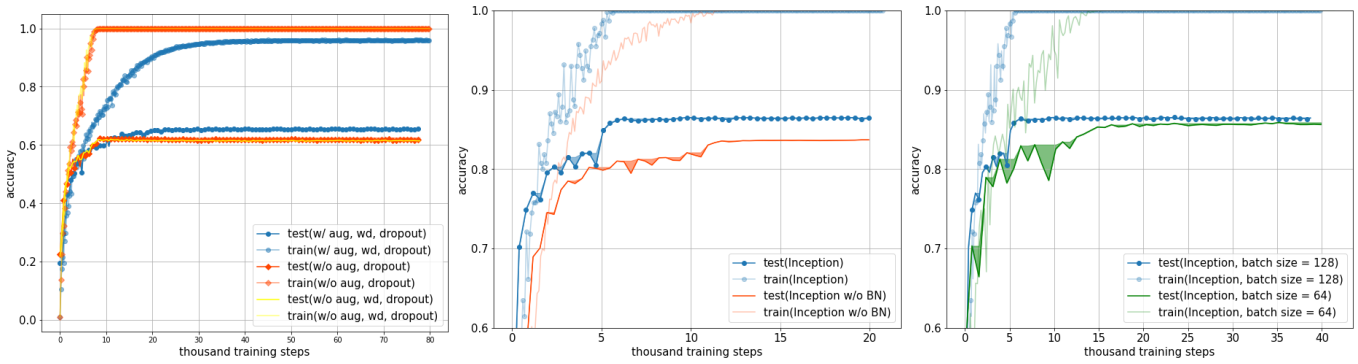
TABLE II: The results of fitting random label

Model	Regularizer	Training Accuracy
Inception	Weight decay	100.0
Alexnet		100.0
MLP 3x512		100.0
MLP 1x512		99.85

The authors conclude that explicit regularizer is not the key of generalization performance, and is neither necessary nor by itself sufficient for controlling generalization error.

IV. Implicit Regularizations

In this section, the authors evaluate the implicitly regularized ability of early stopping and batch normalization (BN)



(a) Inception on Cifar-100 (b) Inception on Cifar-10 (c) Inception on Cifar-10 with different batch sizes

Fig. 3: Effects of implicit regularizers on generalization performance. aug is data augmentation, wd is weight decay, BN is batch normalization. The shaded areas are the cumulative best test accuracy, as an indicator of potential performance gain of early stopping.

on generalization performance. Specifically, the authors test the effect of early stopping of Inception V3 fitting ImageNet [11], and also the effect of early stopping and BN of small Inception on CIFAR10.

A. Reproduce results

Due to computational capacity constraints, instead of training Inception V3 on ImageNet, we train simplified Inception on CIFAR100 in Fig. 3(a). CIFAR100 [12] contains images similar to CIFAR10, having 100 object classes with 600 images annotated per class. We choose CIFAR100 instead of CIFAR10 to mimic the results on ImageNet because it contains more classes of objects.

Training details: For data augmentation, we adopt 3x28x28 random cropping with 4-pixel padding, randomly changing the brightness, contrast, saturation and hue of an image. The brightness, contrast, saturation and hue factors are chosen uniformly from -0.25 to 0.25. The weight decay factor is chosen as 4e-5. The remaining unmentioned experimental details are consistent with Sec. II-A1.

Fig. 3(a) shows the train and test accuracy of Inception on CIFAR100, with the effects of data augmentation, weight decay and dropout. Unlike the case in ImageNet [1], we do not observe any potential benefit of early stopping on CIFAR100. In particular, even when the explicit regulizers are switched off, the impact of early stopping is not remarkable. Besides, again here we have proven the authors' argument that the regulizers are not the fundamental reason for generalization, as the networks continue to perform well after all the regularizers are removed.

Fig. 3(b) shows that early stopping is also not necessarily helpful on CIFAR10. However, it presents the benefits of BN on Inception when fitting CIFAR10. As we can see, networks with BN can converge faster, which is not observed in [1].

B. Discussion

In this section we will focus on discussing the points where our results do not perfectly agree with [1].

Firstly, the authors argue that the early stopping could potentially improve generalization when other regularizers are absent. However, we do not get the same results on CIFAR100. We notice that authors come to the argument using Inception V3 on ImageNet, where both the numbers of data points and network parameters are extremely large (and are of the same order). In our reproduction, we use the simple version of Inception on CIFAR100 (much smaller than ImageNet in both image size and dataset size). Compared with Inception V3 on ImageNet, the simplified Inception is clearly over-parameterized for CIFAR100, which does not necessarily hurt generalization according to the double-descent curve [13]. Naturally, early stopping is more useful when there's obvious overfitting, which is not the case in our reproduction.

Secondly, we observe a phenomenon that BN boosts convergence in Fig. 3(b), which is not observed in [1]. We argue that, this is because we use a larger batch size, which improves the performance of BN, as a sufficient batch size can help BN estimate the batch statistics more accurately [14]. To verify our point, we compare the train and test accuracy curve on CIFAR10, with a batch size of either 64 or 128. From the results in Fig. 3(c), we can see that using BN with a larger batch size can both stabilize the training process (less fluctuation) and boost convergence rate. Apart from increasing convergence rate, we get the same result as [1] that BN indeed provides implicit regularization for the networks and improve generalization performance. However, as indicated in Fig. 3(b), the author's argument that batch normalization stabilizes the training process (with almost no fluctuation) is not held in our reproduction, which might be due to the randomness of SGD optimizer and initialization of network.

V. Conclusion

In this report, we reproduced the results on the experiments of effective capacity of neural networks, impacts of explicit and implicit regularizations on the generalization performance from [1]. Our results verify that the effective capacity of several successful neural network architectures is rich enough to fit the randomly-labeled data. Besides, we validate the authors' argument that either the explicit regularizers or the implicit regulariziers are not the fundamental reason for generalization. The future new research direction could be discovering how structure design of the networks influences generalization ability, instead of focusing on regularization techniques.

APPENDIX

Model Structure The structures of Inception and AlexNet are simplified to adapt to smaller input size in CIFAR10. The detailed architecture of small Inception is shown in Fig. 4.

In order to perfectly reconstruct the small Inception model used in [1], several parameters have to be adjusted from the standard Inception V3:

- Set the `affine` parameter in BatchNorm to be False, so that BatchNorm doesn't introduce more paramters.
- Set the `bias` parameter in the convolutional layers to be True.

With the above adjustments, we are able to reconstruct the small Inception model with exactly the same number of parameters (1,649,402) as [1].

The small AlexNet used in [1] is an AlexNet-style small neural network. It contains two basic modules (5x5 convolution → 3x3 max-pool → local-response-normalization), and two fully-connected layers (384 and 192 hidden units, respectively), followed by a 10-way linear layer for classification. However, the authors didn't provide specific parameters for the convolutional layers or pooling layers (in particular the filter size is not given), so we have to guess some of the parameters such that our small AlexNet has a total number of parameters close to [1].

In our AlexNet model, the convolutional layer always adopts same padding and contains bias term, the stride of max pooling layer is set to 1 (instead of 2 in original AlexNet). The local response normalization layer is set with the same setting as original AlexNet (`size=5, k=2, alpha=1e-4, beta=0.75`). The filter sizes for the first and second convolutional layers are set to be 64 and 256, respectively. Finally, our AlexNet has a total number of parameters 1,375,690, very close to 1,387,786 in [1].

For the MLP 1x512, as its name suggests, it has one hidden layer of 512 hidden units. MLP 3x512, on the other hand, contains three hidden layers, with each layer containing 512 hidden units.

Although the authors of [1] did not explicitly mention whether Batch Normalization was used in AlexNet and MLP in their experiment, we find that in practice, Batch Normalization is necessary for AlexNet and MLP to converge to 0 train error. Therefore, All the models include Batch Normalization layer.

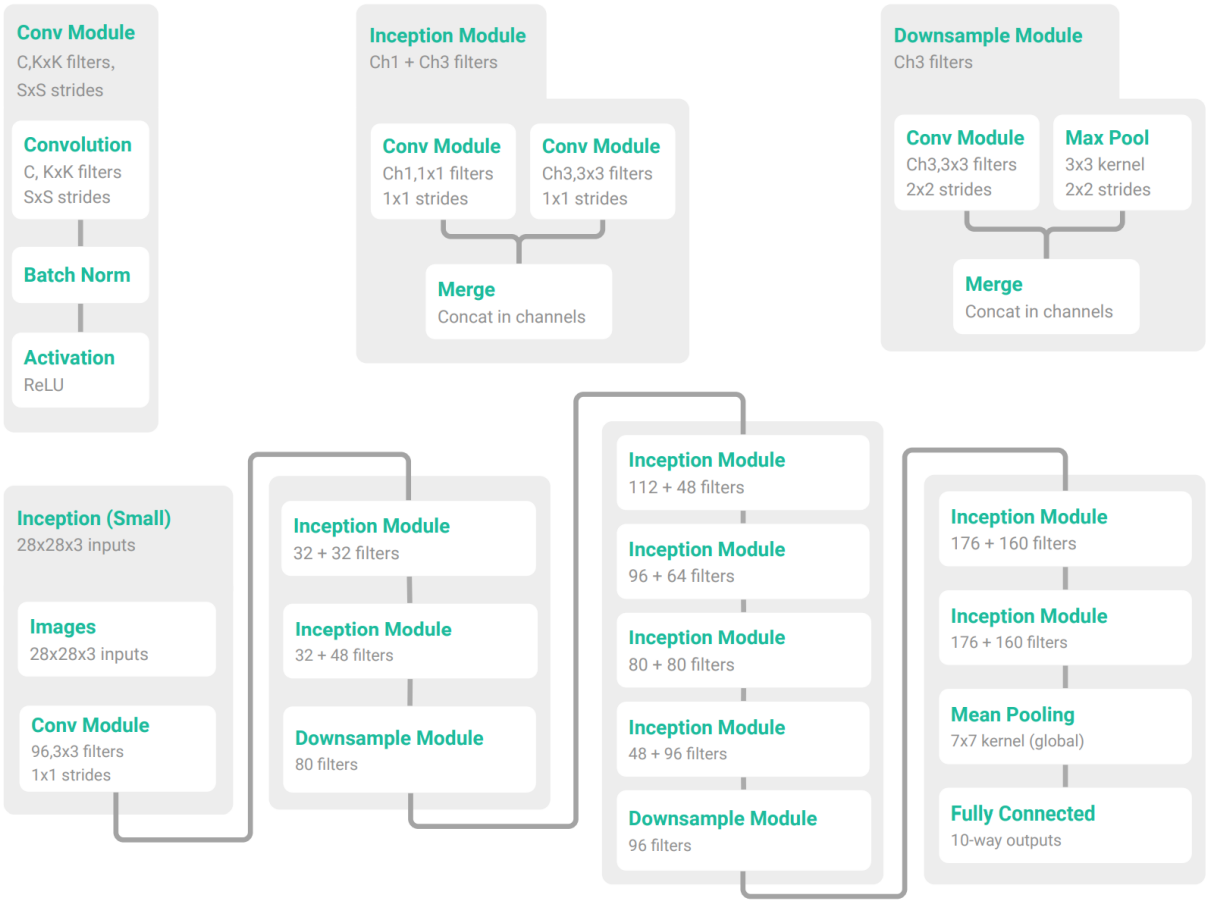


Fig. 4: The architecture of small Inception used for the CIFAR10 dataset. The figures above show the structures of the basic modules and the figure below illustrates the overall structure. Reprinted from [1]

.

REFERENCES

[1] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[2] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.

[3] O. Bousquet and A. Elisseeff, "Stability and generalization," *The Journal of Machine Learning Research*, vol. 2, pp. 499–526, 2002.

[4] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 463–482, 2002.

[5] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[8] L. Rampasek and A. Goldenberg, "Tensorflow: biology's gateway to deep learning?" *Cell systems*, vol. 2, no. 1, pp. 12–14, 2016.

[9] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[10] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[12] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-100 dataset (canadian institute for advanced research)," 2009.

[13] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019.

[14] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.