Multi-Sample Interpolation Training Method

Daojun Liang
Shandong Normal University
Jinan 250014, Shandong Province, China
15154121592, 86
liangdaojun@stu.sdnu.edu.cn

Feng Yang
Shandong Normal University
Jinan 250014, Shandong Province, China
15098738282, 86
yangfeng@sdnu.edu.cn

Xiuping Wang
Shandong Normal University
Jinan 250014, Shandong Province, China
18366133950, 86
wangxiuping@stu.sdnu.edu.cn

ABSTRACT
The mixup training method has achieved a better generalization performance than the traditional training method. But there is no interpretation to why mixup has such a good generalization. In this paper, a series of ablation experiments were first done to prove that the training method of mixup is equivalent to a regularization and data augmentation. Then, we propose several different multi-sample training methods as variations of the mixup, which can also achieve comparable performance with mixup. This method shows that the different mixing methods can achieve the same effect as the original mixup training method. Next, a network architecture that can classify multiple samples at the same time is being proposed. The network architecture prove that mixup does not only learn a linear interpolation between the two categories, but learns to separate the two categories more accurately. Finally, through the fineturning of mixup, the training precision can be further improved.

CCS Concepts
• Computing methodologies→Machine learning→Machine learning approaches→Neural networks.

Keywords
Convolutional Neural Network; Neural Network; Deep Learning

1. INTRODUCTION
Convolutional neural networks have made great progress in many fields, and the research of the network architecture has never stopped. AlexNet [1] is the first to demonstrate the generalization ability of convolutional neural networks on large data. VGGNets [2] show that better performance can be achieved with smaller convolutional kernels and deeper layers. GoogLeNets [3] use different convolution kernels to establish more connections and more diverse representations between adjacent layers. ResNets [4] and Highway Networks [5] add the front layer information to the back layer through the bypass structure, which is more conducive to the backpropagation of the gradient, thus further deepening the depth of the network. ResNeXts [6] combine group convolution into ResNets [4], which perform split-transform-merge operations on features to improve network performance while reducing parameters. DenseNets [7] pass the features of each preceding layer to all of its subsequent layers to alleviate the vanishing/exploding gradient problem and to facilitate information fusion between layers.

In SamplePairing [8] and mixup [9], a data enhancement method using a combination of two pictures in a training set is proposed. SamplePairing [8] randomly selects a sample in the training set to perturb the original sample (adding the two samples and then averaging them) and using the label of the original sample as the label of the new sample. Because different types of training samples are introduced during training, the neural network trained by the method has high training errors and losses, and the original samples need to be used to fine-tune the neural network or use a certain proportion of original samples in each batch.

ICMLT 2018, May 19--21, 2018, JINAN, China

ACM ISBN 978-1-4503-6432-4/18/05…$15.00
DOI: https://doi.org/10.1145/3231884.3231896

2. RELATE WORK
Recently, in SamplePairing [8] and mixup [9], a data augmentation method using a combination of two pictures in a training set is proposed. SamplePairing [8] randomly selects a sample in the training set to perturb the original sample (adding the two samples and then averaging them) and using the label of the original sample as the label of the new sample. Because different types of training samples are introduced during training, the neural network trained by the method has high training errors and losses, and the original samples need to be used to fine-tune the neural network or use a certain proportion of original samples in each batch. The mixup [9] uses a random value from the beta distribution λ as a weight to interpolate the two samples and their corresponding labels, respectively. The neural network trained in this way can achieve relatively low training errors and losses without the need for fine tuning.

There's a lot of work related to regularization. Dropout [10] and DropConnect [11], making each neuron more capable of representation by discarding a certain percentage of neurons or connecting paths during training. Stochastic depth [12] averages architectures with various depths through randomly skipping layers. Swapout samples from abundant set of architectures with dropout and stochastic depth as its special case. In BN [13], the features of each layer in the network will be given a normal distribution again, which makes the neural network easier to learn.

3. MIXUP TRAINING METHOD
In this section, we first introduce the original mixup method in 3.1, and then introduce the more general mixup method in Section 3.2. Finally, in Section 3.3, we introduce the use of mixup for multi-class training.

3.1 The Original Mixup

We found that SamplePairing and mixup are very similar to traditional data enhancement methods, but they are also different. SamplePairing adds other samples from the training set as noise to regularize the neural network during training.

The mixup in the original text is implemented by using a random value to weight the samples and their labels. Mixup can be formalized as:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j \qquad (1) \\ s.t. \quad & 0 \le \lambda \le 1 \end{aligned}$$

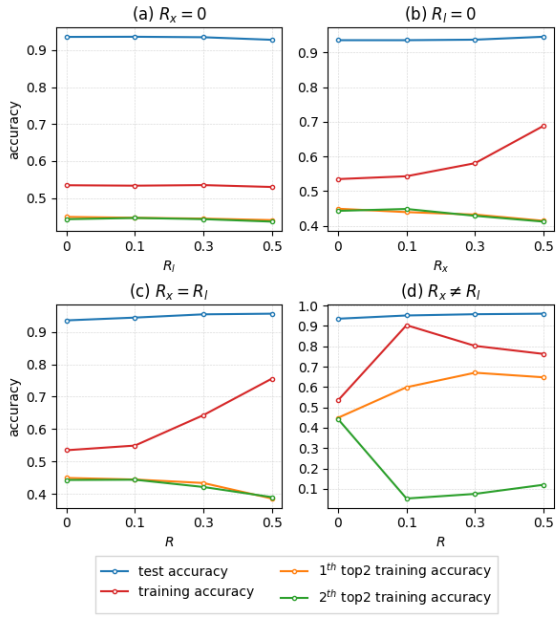

Figure 1. The top2-precision and top2-all-precision means that during training or testing, two samples are mixed in different proportions and then input to the ResNet-56 [4].

where (x_i, y_i) and (x_j, y_j) are two examples drawn at random from our training data, and λ in $[0,1]$. In the mixup, training samples and their labels simultaneously reduce the λ or $1-\lambda$ times, implying this linear relationship. The signal strength becomes λ or $1-\lambda$ times, the corresponding feedback signal will be λ or $1-\lambda$ times that of the original. Our experiments show that training samples and their labels do not simply have a linear relationship.

3.2 More General Mixup

Let's take the sample's mixing ratio as λ_x and the sample label's mixing ratio as λ_l. Our empirical experiments show that the sample mixing ratio λ_x and the label mixing ratio λ_l do not have to be the same and do not have to come from the same distribution. We define the two ratios of λ_x and λ_l to be $[0.5-R_x, 0.5+R_x]$ and $[0.5-R_l, 0.5+R_l]$ respectively, where R_x, R_l in $[0,0.5]$. When R_x is equal to R_l, we write them uniformly as R. Conversely, when we use R, it means R_x is equal to R_l. Figure (1) analyzes the training accuracy of mixup training methods for different values of R_x and R_l. Figure (1.a) shows that when R_l remains unchanged, the training accuracy of the network increases as the defined range of R_x becomes larger. Figure (1.b) shows that when R_x remains unchanged, the training accuracy of the network increases with the increasing range of R_l. But their training precision can not reach 1. Figure (1.c) shows that as R_l, R_x change range becomes larger, the network training accuracy gradually increased. Figure (1.d) shows that R_l and R_x have the same range of variation, but their values are not necessarily the same.

In Figure (1.a), $R_x = 0$, only R_l is changing. We call this training method mixup-L. In Figure (1.b), $R_l = 0$, only R_x is changing, and we call this training method mixup-X. Figure (1.c) to achieve the original paper mixup, then $R_x = R_l$, they simultaneously change. In Figure (1.d), R_x and R_l are all changing, but $R_x \neq R_l$, we call this form mixup-XL.

In Figure (1), top2-precision represents the category for each of the two samples. Consider the confusion between two samples, that is, dividing the two samples into each other's categories, with top2-all-precision equal to top2-precision plus the accuracy of reversing the order of the two categories in top2-precsion. Through the gap between top2-precision and top2-all-precision, we can well measure the degree of confusion that neural networks classify two samples. Note that in the case of multi-label prediction, when the two category labels are the same, since the neural network takes top2 for the prediction, it will inevitably lead to a wrong prediction of one category. We assume that the probabilities of encountering the same category during training and testing are $\frac{1}{N}$, where N is the number of categories. The highest accuracy is $1-\frac{2}{N}$ for the N categories of samples.

Figure (1) shows that the mixup training method is quite different from the traditional training methods. In the traditional training method, the input and output are fixed, the training error can be reduced to 0. In mixup, however, the training error is not reduced to 0 if either end of the inputs and outputs are fixed, but this does not guarantee that the generalization error is higher than the generalization error of the network using the traditional method.

3.3 Multi-Category Classification

These experiments help us to explain why the mixup works so well: it can separate the two categories at the same time. The neural network adjusts to two directions at the same time. When the difference between the two categories is very small (R is

small), the neural network can not distinguish the two categories well, so that the training error can not reach 0. When R are large, the neural network will adjust one of the samples separately. In this way, the neural network will have the opportunity to distinguish the two classes because this process is equivalent to training a sample separately. This is similar to intermittently using SamplePairing for training. In SamplePairing, 2 epochs out of 10 epochs use SamplePairing training and 8 epochs use ERM training. When using SamplePairing training, it is equivalent to the training method when $R \approx 0$ in the mixup. When ERM training is used, it is equivalent to the training method with $R \approx 0.5$ in the mixup, in which the information of one sample almost disappears. Therefore, mixup is a smooth way of SamplePairing, and the mixup training method can be interpreted as: When the mixing ratio is small, mixup is equivalent to the

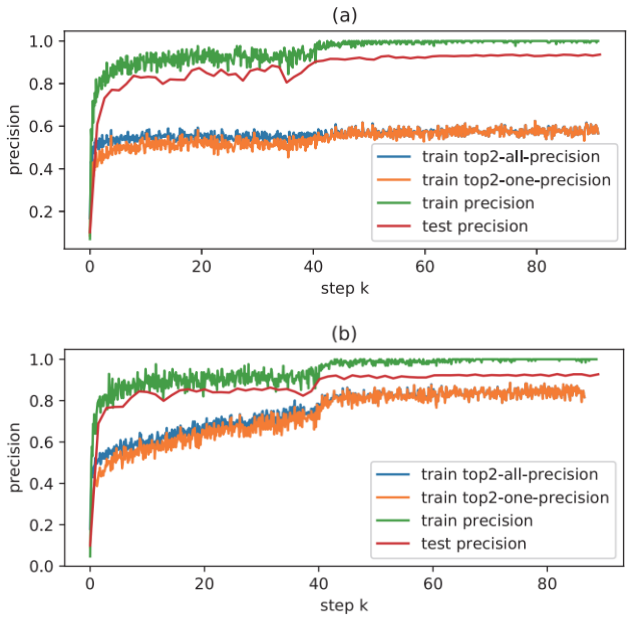

Figure 2. Effect of different mixtures of samples on multi-class network performance. The mixture of the two samples in (a) is additive, and the combination of the two samples in (b) is a concatenation.

regularization process. When the mixing ratio is large, it is similar to the ERM training process.

From Figure (1.c) and Figure (1.d), we can see that the neural network training accuracy is less than 1 when $R \leq 0.5$. This is because the neural network is confused with the sample, that is, each input sample contributes to more than one category simultaneously. The neural network will be confused for samples with overlapping sample types. In order to quantitatively study the degree of confusion, we use the confusion rate to measure the proportion of neural network misclassification. The confusion rate of traditional training methods is 0, and the average training precision is 1. SamplePairing training method mixing rate is 0.5, the average training accuracy is 0.5. When the mixing ratio is $R:(1-R)$, the confusion ratio is $\min(R,1-R)$. For example, if the mixup ratio of a sample is 0.8:0.2, the confusion rate is 0.2, and the average training precision can only reach 0.8. We call the ratio of mixups when R is maximized as the blending ratio of the borders. Obviously, the mixing efficiency of mixups depends on the blending ratio of the boundary. For example, when $R = 0.5$, the boundary blending ratio is 1:0, the confusion rate is 0, and the average training precision is 1.

4. PREDICT MUTILPLE CATEGORIES

4.1 Mixup Variants

To demonstrate that mixup can separate multiple categories simultaneously in a forward process, we have designed experiments that can predict multiple categories simultaneously. In this experiment, all the channels of two images are used as input of neural network, and the probability of top2 of network output is taken as the category of two images respectively. Samples and their labels will use the following mixup form:

$$\begin{aligned} \tilde{x} &= [\lambda x_i \, , (1-\lambda) x_j \,] \\ \tilde{y} &= [\lambda y_i \, , (1-\lambda) y_j \,] \qquad (2) \\ s.t. \quad & 0 \leq \lambda \leq 1 \end{aligned}$$

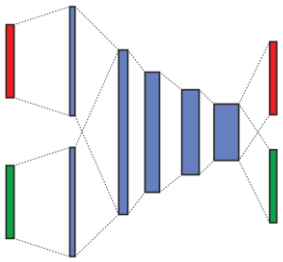

Figure 3. Multi-sample classification network architecture. Red and green represent two samples (input layer) or classification results (output layer), respectively. Green represents the network architecture.

where "[ ]" indicates the concatenation operation. We call this method mixup-C. Although Equation (2) and Equation (1) have the same form, we found that this implementation has a significant impact on the accuracy of one and two categories. Figure (2) shows the training and testing accuracy of top2 and top1 for mixup-C and mixup.

As mentioned in section 3, when two categories of samples are the same, at least one sample is classified as the wrong category. In this experiment, we compare the accuracy of the mixup-C and the mixup. It can be found that there are few interlaced parts in the Figure (2.a) and Figure (2.b). That is, the neural network can accurately classify them into two categories. In Figure (2.b), the neural network predicts the accuracy of the two categories to be 70% at one time, which means that the neural network can separate the two samples into two categories at the same time without any confusion.

4.2 Multi-Sample Confusion Problem

When the neural network classifies two samples at the same time, either the two samples are input into two neural networks respectively, or one sample is predicted first and then the other sample is predicted. How to predict multiple categories simultaneously in a forward process of neural network? Section (3.1) gives a simple way. In this section, we will explore to what extent the neural network confuses two samples, or can we separate them accurately. We not only pay attention to training accuracy, but also focus on test accuracy. To further explore how neural networks can classify two categories simultaneously in a

forward process, while reducing computational time and computational resources. The neural network is designed with two inputs and two outputs. The middle layer of the neural network is designed as a shared structure. The network architecture used in this section is given in Figure (3).

The difference with the traditional network is that the final layer of this network will output the prediction of two categories. We directly add the cross-entropy of the output of the two categories. The loss of the whole network is:

$$Loss = \sum_{i=1}^{N} \lambda_i E_{y_i \sim p_i(y_i)} [\log C_i(y_i)] \quad (3)$$

$$s.t. \quad \sum_{i=1}^{N} \lambda_i = 1$$

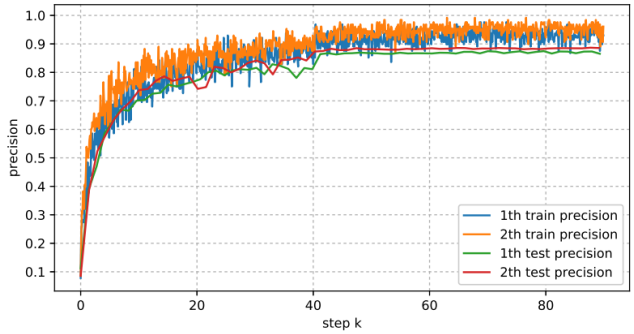

Figure 4. Mixup-C training and testing performance in two categories.

5. EXPERIMENTS

5.1 Training

For comparison purposes, we use the same network architecture for all datasets and set the same hyperparameters and training procedures for neural networks trained on CIFAR-10 [14] dataset. ResNet-56 [4] is used as the basic network architecture. The network architecture is divided into 3 blocks, each of which is a residual block containing two convolutional layers. After each block, the feature size is halved and the number of channels is doubled. ResNet-56 [4] is used for all datasets and set the same hyperparameters and training procedures for neural networks trained on different datasets.

All the networks are trained on two Tesla k80 GPUs using stochastic gradient descent (SGD). We use a weight decay of $1x10^{-4}$ and a Nesterov momentum [15] of 0.9 without dampening. The batch size on each GPU is set to 128 for 100 epochs. The initial learning rate is set to 0.1 and is divided by 10 at 40%, 60% and 80% of the total number of training epochs.

5.2 Mixup-C

Figure (4) shows the training accuracy and test accuracy of the network for two categories on the CIFAR-10 [14] dataset. It can be found that the training accuracy of the two neural networks is about 96%, indicating that there is still some confusion in the neural network. The first category of the average test accuracy of 88.41%, the second category of the average test accuracy is 88.61%, the classification accuracy has been quite good.

This shows that mixup-C can separate two categories at the same time in a forward process. We found some confusion between the first category and the second category, but the degree of confusion is very low. This ensures that the neural network can classify multiple samples.

5.3 Random Interpolation between Samples

When we use the mixup-C network to train two categories of samples, we only change one of the lambdas in equation (2). This means that we can perform random interpolation between samples and its labels.

Figure (5) shows the effect of the two interpolation methods on the multiclass performance of the mixup-C network. We can see that the two types of interpolation have similar performance: one of the two categories is normally trained and the other is close to random guessing. We analyze that this reason arises because the network has more competition among multiple categories when conducting multi-category training. When a category fails to compete in the initial stage of training, the information of the category of the category will not be used in the subsequent training, and all its information is almost completely filtered as noise. Conversely, categories that gain greater competitive advantage at the beginning of training will have a good competitive advantage in the later stages of training. Note that which category will gain competitive advantage at the beginning of the training depends on the value of λ at the beginning. Because the value of λ is subject to a random distribution, a category's competitive victory in the training phase is also random.

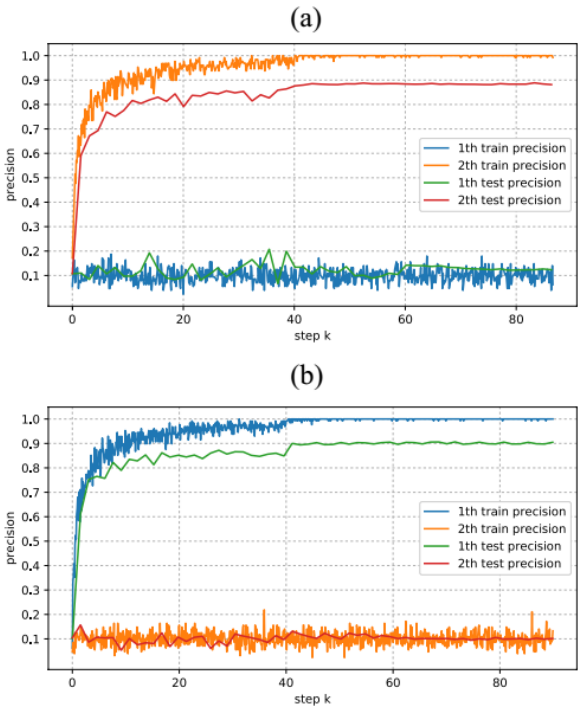

Figure 5. Random interpolation between samples. (a) represents a random interpolation between samples, and (b) represents a random interpolation between the labels.

5.4 Random Label

In order to compare the random interpolation between the samples and the random label for one of the samples, we set a sample from multiple samples as a random label to test the multi-class performance of mixup-C.

Figure (6) shows the performance of mixup-C with random labels. It can be seen that it is very similar to the process of random

interpolation between samples. This proves to some extent the conclusion in Section 5.4 that the information of the category that failed to compete in the initial stage of training will be treated as random noise.

5.5 Fineturning

For networks using different mixup training, further fine-tuning can further improve the generalization performance of the single-category network. Note that in this fine-tuning process, multiple categories of input and output layers will be replaced by a single category.

The results were shown in Table (1). From the Table (1), we can see that the single class precision of the network after the fine-tuning has a greater increase, which is because the confusion between multiple categories is eliminated when the single class is fine-tuning, which makes the network better fit for a single class of samples.

Table 1. Fine-tuning the ResNet-56 using different training methods.

Network	Mixup-X	Mixup-L	Mixup-C
Original	90.2%	90.8%	92.5%
Fine-tuning	93.1%	93.5%	93.7%

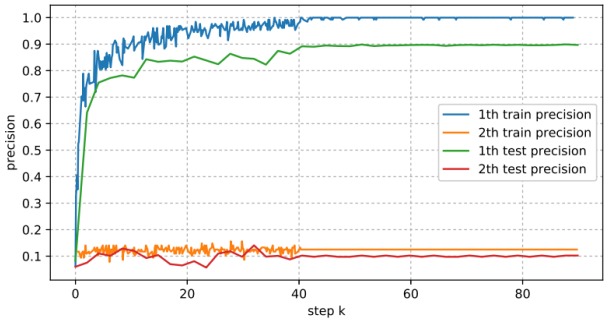

Figure 6. The multiclass classification of mixup-C, where the label of a sample is set to a random value.

6. CONCLUSIONS

In this paper, we analyze the training effect of the mixup method and propose a variation of the mixup, which has the same good performance as the mixup method. Mixup training can use not only random values from the same distribution, but also different random values for samples and its labels, and these random values can come from different distributions. To further understand the mixup approach, we found that the mixup approach can classify multiple categories in a forward process, effectively avoiding sample aliasing. This is also the reason that mixup classification is effective, that is, the neural network can fit the distribution provided by mixup. Based on the mixup method, we can classify multiple categories. We propose a network architecture that can classify multiple categories at the same time in the same forward process. Our experiments show that the architecture has good multi-classification performance. Finally, based on the degree of regularization of the mixup, we find that fine-tuning the neural network trained by mixup can effectively improve the performance of the network.

7. ACKNOWLEDGMENTS

The work is partially supported by the Technology and Development Project of Shandong (No.2013GGX10125).

8. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In International Conference on Neural Information Processing Systems, pages 1097 – 1105, 2012.

[2] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211 – 252, 2014.

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. pages 1 – 9, 2014.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. pages 770 – 778, 2015.

[5] Rupesh Kumar Srivastava, Klaus Greff, and JÃijrgen Schmidhuber. Training very deep networks. Computer Science, 2015.

[6] Saining Xie, Ross Girshick, Piotr DollÃ ar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. 2016.

[7] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. In CVPR, 2016.

[8] Inoue H. Data Augmentation by Pairing Samples for Images Classification. arXiv preprint arXiv:1801.02929, 2018.

[9] H. Zhang, M. Cisse, Y. N. Dauphin, D. Lopez-Paz, mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412, 2017.

[10] J. Ba, B. Frey, Adaptive dropout for training deep neural networks, in: C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26, Curran Associates, Inc.,2013, pp. 3084 – 3092.

[11] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, R. Fergus, Regularization of neural networks using dropconnect, in: S. Dasgupta, D. McAllester (Eds.), Proceedings of the 30th International Conference on Machine Learning, Vol. 28 of Proceedings of Machine Learning Research, PMLR, Atlanta, Georgia, USA, 2013, pp. 1058 – 1066.

[12] G. Huang, Y. Sun, Z. Liu, D. Sedra, K. Q. Weinberger, Deep networks with stochastic depth, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), ComputerVision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 646 – 661.

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. pages 448 – 456, 2015.

[14] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Tech Report, 2009.

[15] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In International Conference on International Conference on Machine Learning, pages III – 1139, 2013.