Phantom Embeddings: Using Embedding Space for Model Regularization in Deep Neural Networks

Mofassir ul Islam Arif[1], Mohsan Jameel[1], Josif Grabocka[2], and Lars Schmidt-Thieme[1]

[1] Information Systems and Machine Learning Lab, University of Hildesheim, Germany
{mofassir,mohsan.jameel,schmidt-thieme}@ismll.uni-hildesheim.de
[2] Department for Computer Science,Albert-Ludwigs-University, Freiburg,Germany
{grabocka}@informatik.uni-freiburg.de

Abstract. The strength of machine learning models stems from their ability to learn complex function approximations from data; however, this strength also makes training deep neural networks challenging. Notably, the complex models tend to memorize the training data, which results in poor regularization performance on test data. The regularization techniques such as L1, L2, dropout, etc. are proposed to reduce the overfitting effect; however, they bring in additional hyperparameters tuning complexity. These methods also fall short when the inter-class similarity is high due to the underlying data distribution, leading to a less accurate model.
In this paper, we present a novel approach to regularize the models by leveraging the information-rich latent embeddings and their high intra-class correlation. We create phantom embeddings from a subset of homogenous samples and use these phantom embeddings to decrease the inter-class similarity of instances in their latent embedding space. The resulting models generalize better as a combination of their embedding, regularizes them without requiring an expensive hyperparameter search. We evaluate our method on two popular and challenging image classification datasets (CIFAR and FashionMNIST) and show how our approach outperforms the standard baselines while displaying better training behavior.

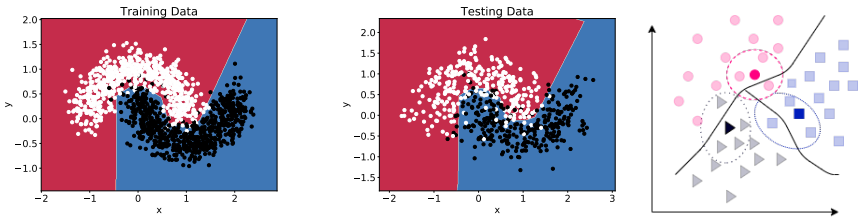
Keywords: Deep Neural Networks · Regulariztaion · Embedding Space.

1 Introduction

The field of computer vision has seen a remarkable increase in capability and complexity in recent years. The use of deep learning models in image classification [10] and object detection [4] tasks have shown a marked increase in their

ability to capture more complex scenarios. Increasingly complex deep learning models such as ResNet [7] and Inception [21] were able to capture more in-depth information from input data. The strength of these deep learning models comes from their ability to take complex data and reduce it to highly expressive latent representations. These latent representations encode an image's spatial information into a vector through repeated convolutions and pooling operations.

Training these complex models bring their challenges. Generally, the true distribution of the data is unknown, and observations are available in a limited number. These models are trained by iteratively minimizing the empirical risk over the training data (also known as Empirical Risk minimization ERM [22]). However, the increasing complexity of the model tends to overfit the data and generalize poorly on the test data, despite using the proper regularization. The theoretical understanding of ERM guarantees convergence as long as the model complexity does not increase with the number of training data [23]. For deep neural networks, an obvious issue arises as the increase in model complexity is not always complemented by an increase in the training data.



(a) Training Data (b) Testing Data (c) Phantom Embeddings

Fig. 1. Fig. 1a shows the overfitted decision boundary on training data. When evaluated on the test set in Fig. 1b the model shows poor generalization, a consequence of overfitting. In Fig. 1c, we show a hypotethical embedding space and the decision boundary created by a deep neural network. The light colors represent the original embeddings while the darker colors represent the phantom embeddings proposed by our method.

To illustrate the aforementioned problems, we train a feed-forward neural network on a synthetic binary classification dataset and visualize the decision boundary in Fig. 1. Fig. 1a shows that the model was able to learn a reasonable decision boundary on the training data. However, due to the limited training examples available to train a complex model, it could not capture a better generalizable decision boundary resulting in poor performance on the test examples, as shown in Fig. 1b. This example showcases two crucial challenges, firstly, how easy it is to overfit and perform poorly on test data. Secondly, in Fig.1a it can be seen that certain instances from differing classes are very close to each other, and ERM fails to provide a procedure to capture those instances.

The model overfitting is treated by introducing the regularization [5, 11, 19, 8] in the ERM objective. However, ERM's problem is most evident around the vicinity of the boundary region, as samples from different classes are in close proximity. Model complexity could be increased to capture these instances, but that violates the convergence guarantee of ERM since the number of instances does not increase with the increase in model complexity. One can mitigate the ERM failure through the Vicinal Risk Minimization Principle [1] by adding a better regularization using data augmentation [16]. Data augmentation mutates the input instances, traditionally through rotating, flipping, and scaling to inject noise in the training data, thereby preventing the model from memorizing it. However, it is limited as it mutates the data within one class vicinity and not across other classes. Other regularization methods involve tunable hyperparameters requiring an expensive configuration search, and the resulting hyperparameters are non-transferable and dataset-specific.

In this paper, we propose a solution for problems stated above by leveraging the latent embeddings to create what we call a 'phantom embedding'. This is done by aggregating the latent embeddings of a subset of the instances from the same class. Using the latent vicinal embedding space allows us to use the information-rich embeddings to inject a hyper-parameter free latent vicinal regularization and boost accuracy. Machine learning models transform input data into their representative embeddings: $\psi : \mathbb{R}^M \rightarrow \mathbb{R}^D$ where M is the original data dimensionality and D is the size of the embedding space. Therefore, by creating this phantom embedding, we create phantom data points to learn on. This is illustrated in in Fig. 1c. This phantom embedding is used to 'pull' the original instance away from the decision boundary and closer to the samples (of the same class) in the embedding space. For the instances already sufficiently away from the decision boundary the 'pull' does not adversely impact since the embedding space is already well seated in the data distribution. We validate on an image classification benchmark task that our proposed solution generalizes better as compared to the existing approaches and achieves higher test accuracies.

Our main contributions include:

- Improvement in classification accuracy by using phantom data points to overcome the base error in a dataset.
- A hyper-parameter free intrinsic regularization to enable training truly deep models.
- Evaluate our model on two popular datasets against established baselines and showcase our performance gains as well as training improvement qualitative and quantitatively.

2 Related Work

Training very deep networks effectively is an open question[20] due to the model complexity. Models with millions of parameters require a lot of data to train effectively, however, millions of training samples are not available for all tasks. A good example of the realistic amount of data needed is [2] with 16M instances.

That is not an option for all machine learning settings especially domains such as medicine [18]. Data augmentation [10] is an efficient method to ensure that data seen by the model is varied during training. Standard augmentation techniques include flipping, scaling, and padding.

Training these models from scratch can be avoided by using the weights of a model that has been trained on a similar dataset and then finetuning the model to fit your need[15] [17]. Transfer learning [15] has enabled training deeper models using a smaller dataset size however, if the goal is complete retraining than the training procedure needs to be adapted to ensure that the model does not memorize the training data.

Methods such as MaxOut [5] add layers into the architecture with a max activation function and have shown to positively impact the convergence behavior when compared to the ReLu activation [14]. DropOut, proposed in [19], addresses the problem of model overfitting by probabilistically turning off neurons in the final embedding layer to create an ensemble of models and has shown to be an effective way to regularize deep neural networks. Similarly in [25], the authors move the regularization from the final layer to the loss layer where they intentionally flip the labels in a mini-batch to ensure that the model generalizes. These methods seek to work on the architecture and loss layer to regularize the model. Methods such as weight decay [11] and batch normalization[8] are aimed at the optimizer and architecture and seek to penalize the weights while training to ensure that models generalize.

In [26] the authors propose the use of taking multiple instances and creating a linear combination of the instances and their label. Sampling from this mixup distribution allows them to learn on fabricated data points.

3 Methodology

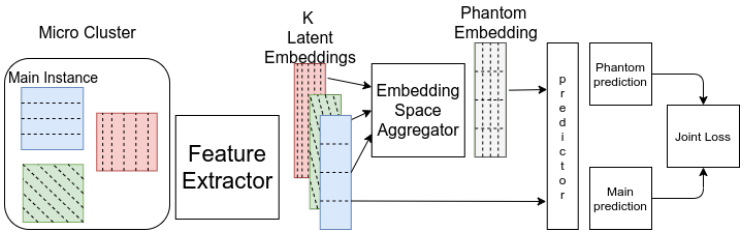


Fig. 2. A training step takes a micro-cluster with K samples, generating K embeddings which are aggregated to create the phantom embedding, This, along with the embedding of the main training instance is passed to the predictor. The combined loss for these predictions is calculated as in Eq. 3.

Consider a machine learning method $\psi(x)$ where x is a dataset sample and $x \in \mathbb{R}^{N \times M}$ corresponds to a multi-category target y where $y \in \{1, \ldots, L\}^N$

among L classes. This model will produce a latent embedding: $\phi : \mathbb{R}^M \rightarrow \mathbb{R}^D$ of the features (the flattened layer after the final convolution block in our case), which is then passed to the prediction layer: $\psi : \mathbb{R}^D \rightarrow \mathbb{R}^L$, for the sake of notational brevity, we will use ϕ and ψ interchangeable with their parameters. The estimated target variable is therefore $\hat{y_n} := \psi(\phi(x_n)), \forall n \in \{1, \ldots, N\}$ and the respective objective function:

$$\arg\min_{\psi,\phi} \sum_{n=1}^{N} \mathcal{L}(y_n, \psi(\phi(x_n))) \tag{1}$$

In this work we propose to make use of the shared similiarities among the instances belonging to the same class and leveraging the collective learned representations of a small subset of instances to generalize the final embedding space. This is done by sampling a 'micro-cluster' of instances belonging to the same class. Note here that 'cluster' is being used in terms of a 'group' and has no relation to the unsupervised clustering methods.

Let us denote the number of instances in each micro-cluster as $K \in \mathbb{N}$ and the number of instances in each respective class as $N_l \in \mathbb{N}, \forall l \in \{1, \ldots, L\}$ therefore for each class it is possible to draw $\binom{N_l}{K}$ many random choices. On these choices, consider, a new dataset transformation $(x, y) \rightarrow (x', y')$, where each element of x' represents a homogeneous cluster from x with K members and each element y' is the respective label of the instances within a homogeneous cluster. Since we are sampling homogenous clusters, $y' = y$. The total number of clusters is defined as $N' = \sum_{l=1}^{L} \binom{N_l}{K}$. The new input features are then $x' \in \mathbb{R}^{N' x K x M}$ and the new targets $y' \in \{1, \ldots, L\}^{N'}$.

This new dataset transformation leads to a model output: $\hat{y_n} := \psi(\phi(x'_{n,k}))$ where $\phi(x'_{n,k})$ is the k^{th} latent embedding and $k \in K$. These K latent embeddings will be used to generalize the final learned embedding by aggregating them as see in Fig. 2. In our proposed approach we use a "Mean Emdedding Space Aggregator" which is explained as: $\phi'(x_n) = \frac{1}{K} \sum_{k=1}^{K} \phi(x'_{n,k})$ where $\phi'(x_n)$ is the phantom embedding from the micro-cluster. The naive approach would be to use this phantom embedding directly in the optimization,resulting in the folllowing objective function:

$$\arg\min_{\phi,\psi} \sum_{n=1}^{N'} \mathcal{L}\Big(y'_n, \psi\big(\frac{1}{K} \sum_{k=1}^{K} \phi(x'_{n,k})\big)\Big) \tag{2}$$

However, Eq. 2 poses a problem since the intra-class variation of challenging datasets can cause the embedding to be too drastically modified, Also, datasets with multi-modal distributions and non-convex hulls can be adversely effected by the naive objective function (Eq. 2) since the micro-cluster can be sampled from the different modes of the data distribution. In its place we propose to use the phantom embedding in the loss function:

$$\mathcal{L} = \alpha \mathcal{L}(y'_n, \psi(\phi(x'_{n,k=0}))) + (1 - \alpha)\mathcal{L}(y'_n, \psi(\phi'(x'_n))) \tag{3}$$

In Eq. 3 we treat the first sample ($k = 0$) as the main instance and the others serve as a guide to improve the embedding space for this instance by 'pulling' the $k = 0^{th}$ towards the phantom embedding. We draw α from the beta distribution and it serves to add stochasticity in the combination of the embeddings and also removes the need for tuning α. Therefore our final objective function is:

$$\arg\min_{\phi,\psi} \sum_{n=1}^{N'} \left[\alpha \mathcal{L}\Big(y'_n, \psi\big(\phi(x'_{n,k=0})\big) \Big) + (1-\alpha) \mathcal{L}\Big(y'_n, \psi\big(\frac{1}{K}\sum_{k=1}^{K} \phi(x'_{n,k})\big) \Big) \right] \quad (4)$$

4 Experiments

In this section, we showcase the results of our approach and compare them with other methods in the domain. All the results presented have been recreated using the original author's provided implementations. These experiments were carried out on NVIDIA 1080Ti, 2080Ti, and V100 GPUs.

4.1 Datasets and Implementation Details

To verify the efficacy of our proposed approach we have chosen two publically available datasets. CIFAR10 [9] and FashionMNIST [24] are popular image classification datasets and are widely used in the computer vision domain for testing new research. They comprise 60000 and 70000 images sized at 32x32 and 28x28 respectively. They offer a challenging problem setting due to the wide intra-class variation and inter-class similarities. Furthermore, these datasets are also easy to overfit the deep convolutional neural networks. Therefore, these datasets provide all the necessary challenges that our work proposes to address.

Our method can be readily included in any machine learning model, for our experiments we have chosen Deep Residual Networks (ResNet-18, ResNet-34, and ResNet-50) as proposed in [7] and as implemented in [13]. The networks under test were initialized as specified in [6] and optimized using Stochastic Gradient Descent(SGD) [12] with batch normalization [8] and a weight decay [11] factor of 0.0005, it should be noted here that the original ResNet architecture used 0.0001. The learning rate was set at 0.1 at the start than the scaled down by a factor of 10 at the 32k and 48k iterations as in [7], training was terminated at 64k iterations. We used a batch size of 128 and the dataset was augmented by padding 4 pixels to the image and translating the image accordingly, the images were also flipped horizontally and normalized by the mean and standard deviation of the entire dataset.

4.2 Results

In this section we evaluate our model by answering the following research question:.

1. **RQ1**: Can classification accuracy be improved by creating a phantom embedding for data points?

2. **RQ2**: Can a better embedding space lead to a more robust model?
3. **RQ3**: Can we add intrinsic regularization by using the embedding space direclty?

4.3 RQ1: Classification Accuracy

The baselines were chosen based on their relevance to the approach that we have outlined in this paper. We have used the DisturbLabel [25] as implemented in [3], ResNet with Dropout [19] and we also compare against the vanilla variants of the ResNet architectures. DisturbLabel seeks to regularize the loss layer rather than the parameters and DropOut seeks to create an inherent ensemble of neural networks by stochastically turning off a certain amount neurons in the embedding layer to prevent the models from learning the training data. A comparison of our method to the baselines can be seen in Tab. 1.

Table 1. Classification Accuracy on CIFAR using ResNet-18 architecture. We report the final accuracy as **Acc** and also the **Mean** and **Max** accuracies for the last 5 epochs to illustrate training stability towards convergence.

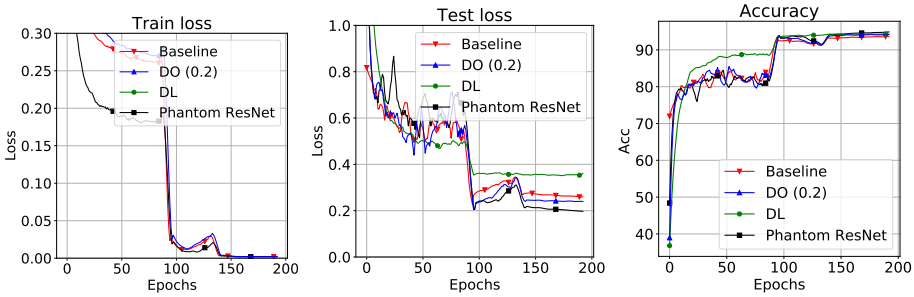
	Accuracy		
Method	**Acc**	**Mean Acc**	**Max**
ResNet18	93.5	93.68	93.68
ResNet18 Dropout	94.11	94.09	94.21
ResNet18 DisturbLabel	94.2	94.28	94.33
Phantom ResNet18	**94.91**	**94.84**	**94.91**

It can be seen in Tab. 1 and 2 that our proposed method is performing better than all the baselines in terms of the accuracy, however, it should also be noted that the overall variance in the results at the time of convergence is also better than the baselines.

Table 2. Classification Accuracy on CIFAR using ResNet34

	Accuracy		
Method	**Acc**	**Mean**	**Max**
ResNet34	93.65	93.71	93.79
ResNet34 Dropout	93.92	93.97	94.03
ResNet34 DisturbLabel	93.73	93.79	93.81
Phantom ResNet34	**94.52**	**94.52**	**94.6**

For Phantom ResNet, we see a 1.47% and 0.88% gain for ResNet-18 and ResNet-34 accuracies respectively. The decrease in the overall 'performance gain' when moving from ResNet-18 to ResNet-34 can be attributed to the fact that



(a) Train loss (b) Test loss (c) Test Accuracy

Fig. 3. ResNet-18 Training and testing behaviors: **Baseline** refers to the original network baseline while **DO** and **DL** refer to the DropOut and DisturbLabel baselines.

ResNet-34 is a more complex model. ResNet-18 has 0.27M parameters while ResNet-34 has 0.46M, so by doubling the parameters of the network we expect a more expressive model that already improves upon the shortcomings of the former. A more important trend in Tab. 2 is the behavior of ResNet34 Dropout and ResNet34 DisturbLabel values for which we only see an improvement over ResNet34 of 0.27% and 0.1% respectively. In Tab. 1, for ResNet18 Dropout and ResNet18 DisturbLabel we saw an improvement of 0.61% and 0.7% over the vanilla ResNet18. It can be see that the Dropout and DisturbLabel, while still better than the vanilla ResNet lose a significant amount of their gains when the model parameters double from ResNet18 to ResNet34 i-e model complexity increases. These methods do not take into account the highly similar embeddings of data points from different classes during optimization and thus, suffer in final accuracies. Our method uses the latent representation from multiple instances of a class to regularize the model and prevent the highly similar data points from different classes from being too close to the decision boundary.

Table 3. Classification Accuracy on FashionMNIST

	Accuracy	
Method	**ResNet-18**	**ResNet-34**
ResNet	94.78	94.93
ResNet-Dropout	94.97	95.11
ResNet-Disturb	94.95	94.97
Phantom ResNet	**95.07**	**95.38**

In Tab. 3 we can see that the results for our approach continue to outperform the baselines on the FashionMNIST dataset which comes with its own

(a) Train loss (b) Test loss (c) Test Accuracy

Fig. 4. ResNet-34 Training and testing behaviors: With a more complex network, our method continues to outperform the baselines.

set of challenges since the images are now 28x28 and comprise of a single channel rather than the standard RGB channels of CIFAR.

Consistent accuracy improvement across these datasets and over varying architecture complexities shows that our method is robust enough to deal with a wide variety of scenarios. Furthermore, it should be pointed out that the accuracies for the baselines required a large hyper-parameter search to get to these values whereas our proposal required no such search for performance.

4.4 RQ2: Robustness

A sufficiently well-trained algorithm should be able to reduce the error on the test set, the reduction of test error is inextricably tied to the training process. Our proposed methods seeks to mitigate overfitting by enriching the embedding space ensuring that the model generalizes well thus preventing errors in similar classes. It can be seen in Fig. 3b that our model is converging to a lower Test loss, this is an outcome of the enriched embedding space that actively helps optimize the model to learn a more general representation from the training data. The outcome of this approach reflects readily in Tab. 1 in the final accuracies, furthermore considering Fig. 3c it can be seen that our proposed model takes a more deliberative approach in the initial learning stage up to the first 100 epochs. While other models are shooting up quickly in accuracy values, and then later failing to maintain their lead, our approach focuses on learning better representations and penalizing itself when it doesn't more aggressively in order to arrive at the better final optimal network weights.

The same trend is observed when training ResNet-34 as shown in Fig. 4. The only difference being that models not trained with inherent embedding space enrichment in mind suffer more due to the higher complexity of the underlying networks. In both Fig. 3 and Fig. 4 it can be seen that ResNet-Dropout seems to be more stable in terms of its fluctuations during the middle of the training process, between epoch 100 and 150, however it still fails to match our method

Table 4. Classification Accuracy on CIFAR using ResNet50

	Accuracy		
Method	**Acc**	**Mean**	**Max**
ResNet50	93.86	93.25	93.34
ResNet50 Dropout	93.21	93.17	93.25
ResNet50 DisturbLabel	94.37	94.352	94.38
Phantom ResNet50	**94.48**	**94.54**	**94.71**

in the final loss as well as final accuracy. This highlights the problems laid out in the introduction section where a model loses on accuracy in an attempt to not overfit.

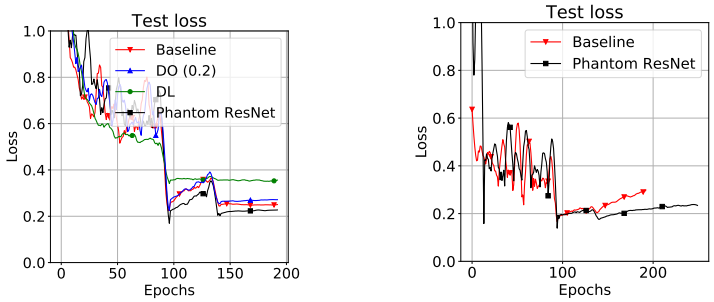
4.5 RQ3: Intrinsic Regularization

As stated earlier, training deep models are hampered by the model memorizing the training data and then showing poor performance on the test data. This problem comes to the forefront when dealing with a truly deep model like ResNet-50 which comes with 0.88M trainable parameters. Training such a model from scratch requires an immense amount of data or a clever regularization scheme. The scheme needs to be searched for over several runs and hyper-parameter configurations. This is a time-consuming and expensive procedure since training ResNet-50 can take up to 7-11 hours on a modern GPU. Our proposed method allows for the data samples to contribute not just to the learning but to the regularization as well, Tab. 4. By intrinsically learning the regularization with the help of similar images and generalizing the weights of our embedding layer with our proposed phantom embeddings we are able to regularize the model as it trains. This behavior is on display in Fig. 5 where it can be seen that our model is leading to a marked lower test loss while the baseline models struggle to match its performance. Given enough time (days) an ideal configuration for the baselines could be arrived to match the performance of our model however, our model provides it without the need for the extensive search required by the baselines.

In Fig. 5b we intentionally allowed the models to run past their convergence point to see how the baseline and our model handle such cases. It can be seen that the baselines runs off and starts to overfit, leading to an increasing test loss while our method shows a noticably better performance and maintains a lower test loss.

4.6 Ablation Study

In order to showcase the effect of different numbers of samples from the same class (K) we varied K from 1 (baseline) to 7 and in Tab. 5. It was seen that while increasing K led to increasing performance over the baselines, the percentage gain vs model complexity didn't justify the use of higher K. All the results reported have been therefore conducted with $K = 2$



(a) CIFAR10 (b) FashionMNIST

Fig. 5. Intrinsic Regularization in ResNet50: The phantom embeddings prevent overfitting even when the training regime is specifically aiming to overfit.

Table 5. Ablation Study: Investigating the effect of increasing K on the classification accuracy.

	Accuracy			
Model	K=1	K=2	K=3	K=4
ResNet 18	93.5	94.91	94.01	93.9
ResNet 34	93.65	94.58	94.3	94.2

5 Conclusion

In this paper, we have shown how embedding spaces can be directly used to regularize deeper neural networks by creating phantom embeddings around the true data points by aggregating the embeddings together and then optimizing the model with the phantom embedding as a co-target. We have shown how our method outperforms the baselines two famous and competitive datasets. Our method also introduces an intrinsic regularization which enables us to train deeper models without an extensive hyper-parameter search.

References

1. Chapelle, O., Weston, J., Bottou, L., Vapnik, V.: Vicinal risk minimization. In: Advances in neural information processing systems. pp. 416–422 (2001)
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
3. Farzaneh, A.: Disturblabel-pytorch (2019), https://github.com/amirhfarzaneh/disturblabel-pytorch
4. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
5. Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. In: International conference on machine learning. pp. 1319–1327 (2013)

6. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. pp. 1026–1034 (2015)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
8. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
9. Krizhevsky, A., Nair, V., Hinton, G.: The cifar-10 dataset. online: http://www. cs. toronto. edu/kriz/cifar. html **55** (2014)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
11. Krogh, A., Hertz, J.A.: A simple weight decay can improve generalization. In: Advances in neural information processing systems. pp. 950–957 (1992)
12. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation **1**(4), 541–551 (1989)
13. Li, K.: kuangliu/pytorch-cifar (2017), https://github.com/kuangliu/pytorch-cifar
14. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
15. Pratt, L.Y.: Discriminability-based transfer between neural networks. In: Advances in neural information processing systems. pp. 204–211 (1993)
16. Simard, P.Y., LeCun, Y.A., Denker, J.S., Victorri, B.: Transformation invariance in pattern recognitiontangent distance and tangent propagation. In: Neural networks: tricks of the trade, pp. 239–274. Springer (1998)
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
18. Sirinukunwattana, K., Raza, S.E.A., Tsang, Y.W., Snead, D.R., Cree, I.A., Rajpoot, N.M.: Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. IEEE transactions on medical imaging **35**(5), 1196–1206 (2016)
19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
20. Srivastava, R.K., Greff, K., Schmidhuber, J.: Training very deep networks. In: Advances in neural information processing systems. pp. 2377–2385 (2015)
21. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
22. Vapnik, V., Vapnik, V.: Statistical learning theory wiley. New York **1** (1998)
23. Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. In: Measures of complexity, pp. 11–30. Springer (2015)
24. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
25. Xie, L., Wang, J., Wei, Z., Wang, M., Tian, Q.: Disturblabel: Regularizing cnn on the loss layer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4753–4762 (2016)
26. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)