AL2: PROGRESSIVE ACTIVATION LOSS FOR LEARNING GENERAL REPRESENTATIONS IN CLASSIFICATION NEURAL NETWORKS

Majed El Helou *Frederike Dümbgen* *Sabine Süsstrunk*

School of Computer and Communication Sciences, EPFL, Switzerland.

ABSTRACT

The large capacity of neural networks enables them to learn complex functions. To avoid overfitting, networks however require a lot of training data that can be expensive and time-consuming to collect. A common practical approach to attenuate overfitting is the use of network regularization techniques.

We propose a novel regularization method that progressively penalizes the magnitude of activations during training. The combined activation signals produced by all neurons in a given layer form the representation of the input image in that feature space. We propose to regularize this representation in the last feature layer before classification layers. Our method's effect on generalization is analyzed with label randomization tests and cumulative ablations. Experimental results show the advantages of our approach in comparison with commonly-used regularizers on standard benchmark datasets.

***Index Terms*—** Neural network, feature representation, regularization, generalization, overfitting.

1. INTRODUCTION

Deep neural networks continue to achieve increasingly-better results on a wide range of tasks; medical image analysis [1], semantic segmentation [2], finding robust features for audiovisual emotion recognition and object recognition [3]. Improvements in the underlying hardware and in parallelization strategies [4, 5] pave the way for ever larger networks. The size of such networks contributes to the complexity of the functions they can model and thus allows the learning of richer representations. However, this increase in complexity can also come at a cost. The capacity of deeper networks increases and so does their potential to memorize [6]. This problem drives the need for larger and more varied training datasets. Such datasets increase the training time, and are also expensive and time-consuming to collect.

To reduce overfitting and improve generalization, various regularization methods are currently used in the training of neural networks. Regularizers such as batch normalization [7], dropout [8], and weight decay [9] are commonly used but are not sufficient [10], and the neural networks are still capable of simply memorizing an entire training set [11]. Neural network regularization remains an open problem [10, 11, 12, 13]. A recent method addresses this problem by proposing to minimize the intra-class entropy of the network representations [10]. However, the main assumption of intra-class similarity fails in the presence of incorrect labels, again requiring the costly perfectly-annotated training datasets.

We propose a simple regularization method that is applied on the feature representation learned by the neural network. This is the representation used by the final linear layers for classification. Inspired by recent findings that neural networks learn general patterns first [6, 14, 15], we propose an ℓ_2-based activation-regularization loss (AL2) that increases per epoch to progressively regularize the network and not allow it to memorize the dataset-specific patterns that lead to overfitting. AL2 directly acts on feature representations through a loss imposed on the magnitude of their activations.

Label randomization results show that our AL2 regularization significantly improves the generalization of the baseline convolutional neural network (CNN). AL2 has a significant effect on the fundamental representation learning, as shown by our canonical correlation analysis [16, 17] in Section 5.1. Additionally, we show that our method combines well with batch normalization, dropout, and weight decay, which can thus achieve better generalization when combined with AL2. Besides label randomization, the cumulative ablation study [12] results in Section 5.2 show that our CNN trained with AL2 has better generalization strength than the different baselines, even at epochs where both have roughly equal test accuracy.

Our contributions are summarized as follows. **1)** We present AL2, a progressive regularization method acting on the activations of the feature representation learned by neural networks before their final classification layers. **2)** We show that our approach improves the generalization of the learned representation: first empirically with label randomization experiments, then using a recent cumulative ablation strategy for assessing the generalization of learned representations. **3)** We analyze the effect of AL2 on the learned representation through a canonical correlation analysis.

2. RELATED WORK

Generalization. Generalization in neural networks remains an open question [18, 19]. The sharpness or flatness of the minima found in weight optimization is commonly used to indicate, respectively, bad or good generalization [20, 21]. However, this belief is undermined by the fact that, for different flatness definitions, the value of flatness can be modified without modifying the function learned by the neural network [22]. Even the performance in terms of error on the held-out validation or test sets is not always a perfect indicator of generalization [23]. One approach to assess the quality of the feature representation learned by a network is to evaluate how much it actually memorizes. This can be achieved by training with a portion of randomized class labels, as the only way the network can learn to predict these random labels is by memorizing this data [11, 24]. The result is a measure related to the empirical Rademacher complexity [25].

Recently, learning despite the presence of corrupt labels in the dataset has become popular [26, 27, 28]. These methods, however,

explicitly aim to solve this noisy-learning problem, by modeling it or by re-labeling the dataset. This is not our objective in using corrupt labels. We use the randomization as an assessment tool of the effects of regularizers on memorization. Another assessment method we use consists of randomly ablating activation signals at inference time, and its results correlate well with generalization strength [12].

Regularization. The most commonly-used regularization methods to reduce network overfitting are batch normalization [7], dropout [8], and weight decay [9]. Batch normalization attempts to stabilize the output of one layer to aid the learning of the following one, dropout attempts to increase robustness by forcing random signal ablations during training, and weight decay reduces network complexity by penalizing the norm of some or all optimization weights. It is recently shown that batch normalization and dropout have disharmonious behaviors [13], as they have opposite effects on feature variance between training and inference. Network regularization remains an open problem [10, 11, 12], along with the study of generalization.

Representations. Feature representations are not only important for transfer learning but also for application-specific feature extraction [1, 29]. Canonical correlation, which we use in our representation analysis, has been recently shown to be a good distance metric to measure similarities of learned representations and to obtain more insights [16, 17].

3. METHOD

We present a regularization method that can be applied on standard classification neural networks. The network architecture is first separated into a *trunk* ϕ and a *head* ψ. The trunk extracts features from the input image and creates a representation signal that is passed to the head. The head then uses the extracted features to perform its classification and predict a probability for each class. The representation learned by the trunk should focus on important image features that can generalize well to unseen data, and not simply extract data-specific patterns. It is this representation that we regularize using our AL2 loss.

The regularization loss is the norm of the feature layer's activation values, and is added as an auxiliary loss term to the classification loss. The overall loss for mini-batch $\mathcal{B}$ is then given at epoch e by

$$\mathcal{L}_e\big(x, y; \Theta\big) = \sum_{x \in \mathcal{B}} \mathcal{L}_c\big(\psi(\phi(x)), y; \Theta_c\big) + \lambda_e \mathcal{L}_r\big(\phi(x); \Theta_r\big), \quad (1)$$

where (x, y) are (image, label) pairs, $\Theta = \Theta_c \cup \Theta_r$ is the set of parameters over which the loss is optimized, and e is a given epoch in the training. $\mathcal{L}_c$ is the classification loss, $\mathcal{L}_r$ is our activation regularization loss, $\phi(\cdot)$ is the function learned by the trunk to extract the feature representation, $\psi(\cdot)$ is the function learned by the head to perform the prediction, and λ is a series of weights. In all our experiments, $\mathcal{L}_c$ is the cross-entropy loss, $\mathcal{L}_r$ is the ℓ_2 norm and the series of weights λ is defined as

$$\lambda_e = \lambda_{e-1} * (1.1 * u[5 - \lambda_{e-1}] + 1.01 * u[\lambda_{e-1} - 5]) \quad (2)$$

$\forall e > 0$, where $u[\cdot]$ is the Heaviside function. Results are not extremely sensitive to changes in this series of empirically-chosen weights as long as they are increasing with an exponential trend, even when a geometric series (only a single factor) is used. We thus use this series of weight values with $\lambda_0 = 0.01$ in all our experiments. Similar to weight decay or other regularizers, the parameter λ can be tweaked for a given dataset or network architecture. The reason it is progressively increasing is that neural networks learn general patterns first, and then overfit to the data-specific patterns [6, 14, 15]. Therefore, by leaving less and less flexibility to the network as the training advances, we limit its memorization capacity in later stages and minimally affect its learning phase in the earlier stages.

Our auxiliary regularization loss does not directly constrain a set of weights, whether in the trunk or in the head of the network. It only constrains the activations of the learned representation. This makes it more general than weight decay, which, in contrast, directly acts on a user-specified set of weights. In fact, weight decay has, in our experiments, the least effect on the trunk's final activation magnitudes when compared with the other regularizers. Our AL2 does, however, regularize the network, but while leaving the flexibility to use any or all of the trunk's layers to minimize this loss.

4. LABEL RANDOMIZATION EXPERIMENTS

We evaluate our method on a VGG-like (2D convolution, maxpooling, ReLU, and linear layers) CNN architecture[1], designed to examine the effects of different regularizers. For reproducibility purposes, all the details of the network architecture and the training settings are presented in the supplementary material[2]. For weight decay, which is sensitive to its chosen weight, we run a parameter search and find the best weight decay value of $5 \times 1e-4$, which is also a value typically used in practice. This value gives the best performance for weight decay without AL2.

We carry out the evaluation of network memorization with label randomization experiments [11, 24]. For each training dataset, a fixed percentage of labels is corrupted with labels chosen uniformly at random from the set of incorrect labels for a given training image (symmetric label noise). We then train the baseline (bare) network with no regularization, the network with batch normalization (BN), with dropout (DO) or with weight decay (WD) on the same corrupt dataset and starting from the same weight initialization. We repeat the training of each of these four networks with our AL2 regularization, again with the same corrupt dataset and starting from the same weight initialization. Results are reported in Table 1 for the MNIST dataset and for 75% corrupt random labels. Further results on MNIST, Fashion-MNIST, and CIFAR10 each with 75%, 50%, 25%, and 0% corrupt random labels are additionally provided in the supplementary material, totaling 96 different networks trained for 700 epochs each.

The results in Table 1 show the test accuracy (TA), the cross-entropy loss $\mathcal{L}_c$ and our regularization loss $\mathcal{L}_r$ at different epochs during the network training. We see that using AL2 significantly improves the generalization of the network assessed at the final epoch, by limiting the overfitting through regularization. The test accuracy improvement is of 60 percentage points in the most extreme case (with weight decay, Table 1). Without using AL2 during training, the best performance is obtained when using dropout. With dropout, compared to other network configurations, we note one interesting phenomenon. The network trained with dropout regularization also indirectly minimizes $\mathcal{L}_r$, an order of magnitude smaller than with

[1] https://github.com/majedelhelou/AL2

[2] https://infoscience.epfl.ch/record/271444

Different metrics evaluated across training epochs (without/with AL2)								
Baseline	Metric	epoch=100	epoch=200	epoch=300	epoch=400	epoch=500	epoch=600	epoch=700
Bare	TA	84.20/95.25	45.30/94.92	25.25/93.07	23.83/88.76	26.07/79.64	26.45/75.88	25.84/**68.46**
	$\mathcal{L}_c$	2.15/2.22	1.78/2.19	0.89/2.15	0.19/2.11	0.04/2.08	0.01/2.07	0.00/2.08
	$\mathcal{L}_r$	3.20/0.24	10.93/0.10	26.12/0.06	54.42/0.03	74.49/0.02	103.26/0.01	119.10/0.00
BN [7]	TA	74.72/95.47	36.65/94.48	26.72/90.20	25.97/85.34	25.88/83.02	25.60/81.53	25.55/**81.16**
	$\mathcal{L}_c$	2.07/2.22	1.48/2.19	0.30/2.15	0.04/2.12	0.01/2.11	0.01/2.12	0.01/2.14
	$\mathcal{L}_r$	0.84/0.24	2.35/0.10	6.46/0.06	9.25/0.03	10.40/0.01	11.06/0.01	11.51/0.00
DO [8]	TA	96.13/94.43	96.47/95.03	95.93/95.03	92.74/94.79	81.96/92.15	68.12/92.69	55.39/**91.70**
	$\mathcal{L}_c$	2.22/2.23	2.20/2.22	2.17/2.20	2.13/2.20	2.05/2.20	1.94/2.21	1.79/2.23
	$\mathcal{L}_r$	0.26/0.24	0.30/0.09	0.41/0.04	0.61/0.02	1.00/0.01	1.50/0.00	1.92/0.00
WD [9]	TA	88.91/95.21	50.87/95.47	27.98/95.17	27.66/94.03	25.14/91.42	28.05/89.81	25.57/**86.98**
	$\mathcal{L}_c$	2.16/2.22	1.87/2.20	1.06/2.18	0.32/2.16	0.07/2.16	0.04/2.17	0.02/2.19
	$\mathcal{L}_r$	2.94/0.23	10.52/0.09	26.04/0.05	53.65/0.02	81.53/0.01	84.64/0.00	107.80/0.00

Table 1: Test accuracy (TA), training cross-entropy loss $\mathcal{L}_c$, and our training regularization loss $\mathcal{L}_r$ which is shown for AL2 multiplied by 100 for readability. We evaluate all metrics at different epochs and with different baselines (no regularization Bare, batch normalization BN [7], dropout DO [8], and weight decay WD [9]), without/with AL2. The networks are trained on the MNIST dataset with 75% corrupt labels.

Area under cumulative ablation curve (/100) evaluated across training epochs (without/with AL2)							
Baseline	epoch=100	epoch=200	epoch=300	epoch=400	epoch=500	epoch=600	epoch=700
Bare	35.44/77.81	19.17/72.67	15.52/69.44	14.73/64.11	15.36/55.08	15.36/51.73	15.19/**47.65**
BN [7]	35.08/77.01	19.17/71.23	15.80/63.48	15.79/57.42	15.64/55.67	15.69/54.97	15.60/**54.96**
DO [8]	81.66/78.52	79.90/78.74	76.23/78.80	70.38/78.31	60.17/73.57	49.86/73.30	41.39/**71.61**
WD [9]	39.50/78.18	20.74/74.83	15.94/74.39	16.09/72.97	15.40/67.62	16.12/64.85	15.35/**62.63**

Table 2: We evaluate the area under the cumulative ablation curve at different epochs and with different baselines, without/with AL2. The networks are trained on the MNIST dataset with 75% corrupt labels.

batch normalization, and two order of magnitudes smaller than with weight decay or the bare network. We thus notice that dropout tends to lead to a smaller $\mathcal{L}_r$, which AL2 explicitly penalizes to a much larger degree. Counter-intuitively, weight decay hardly decreases the magnitude of the activations in the final feature representation layer that is created by the trunk of the network. This underlines the different effects obtained by regularizing activations or regularizing network weights as done by weight decay.

These observations and insights on experiments with no corrupt labels are discussed in more detail in the supplementary material. We also note here that on 75% corrupt data, the bare baseline achieves a test accuracy of 25.84% but of 68.46% with AL2, while the training cross-entropy loss is non-zero for the AL2-regularized network. This indicates that we could correct the labels by re-labeling the dataset with the AL2-regularized network then repeat the training to improve the performance. However, our objective is not to classify data with noisy-label training, but rather to use the randomized label tests to assess network generalization against memorization strength, for different regularizers.

5. NEURAL REPRESENTATION ANALYSIS

5.1. Representation analysis with canonical correlation

The representation learned by a neural network depicts how different neurons respond to the given input data, in terms of their activation values. We feed forward a data point to the network and collect the activations of all neurons in a given layer into a vector in $\mathbb{R}^a$. Collecting and grouping such vectors for n different data points yields the matrix $R_1 \in \mathbb{R}^{a \times n}$, which holds the representation of that set of data points by the neural network. Recent methods have proposed to use canonical correlation by computing a similarity metric from the series of correlation coefficients, which we briefly review in what follows.

The canonical correlation coefficient ρ for two matrices $R_1 \in \mathbb{R}^{a \times n}$ and $R_2 \in \mathbb{R}^{b \times n}$ is given by

$$\rho = \underset{(\omega_1, \omega_2) \in (\mathbb{R}^a, \mathbb{R}^b)}{max} \left(\frac{\langle \omega_1^T R_1, \omega_2^T R_2 \rangle}{||\omega_1^T R_1|| \cdot ||\omega_2^T R_2||} \right), \tag{3}$$

and the corresponding canonical correlation directions are $\omega_1^T R_1$ and $\omega_2^T R_2$. ρ measures the degree of correlation between these two direction vectors. One can solve for the next-best ρ value, which corresponds to two new direction vectors $\omega_1^T R_1$ and $\omega_2^T R_2$ that are respectively orthogonal to the corresponding first two vectors. Repeating this process, with each vector of the new couple ($\omega_1^T R_1$, $\omega_2^T R_2$) being orthogonal to the vector space spanned by the corresponding previously-found direction vectors, yields a sequence of ρ values of size $min(a, b)$. These coefficient values are indicative of the similarity between R_1 and R_2. The larger the values are, the more similar are R_1 and R_2.

Both SVCCA [16] and PWCCA [17] compute weighted averages of the canonical correlation coefficients. To avoid any loss of

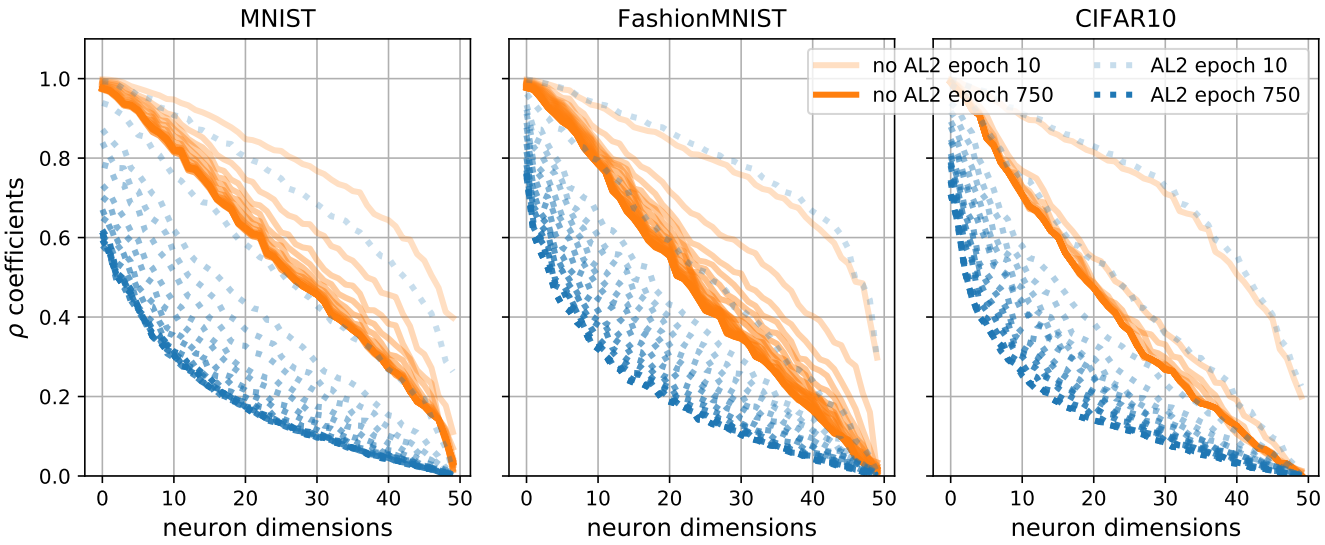

Fig. 1: Canonical correlation coefficients ρ as a function of neuron dimensions between the learned feature representation $\phi(x)$ at the beginning of training and at given training epochs ranging from 10 to 750 (illustrated with increasing color intensities), for the three datasets MNIST, FashionMNIST and CIFAR10. The plots show that AL2 has a significant effect on the representation learning process, confirming the fundamentally different classification results reported in Table 1. Results are obtained with a training dropout rate of 50 percent and for the first batch of each dataset. Both networks are initialized with identical weight values for a fair comparison. Best viewed on screen.

information through averaging, we visualize the entire sequences of ρ coefficients in our analysis (Fig. 1).

For each dataset, we obtain a representative sample of correlation coefficients by passing the first random training batch through the network. We form a matrix of shape $50 \times 16{,}000$ for MNIST and FashionMNIST and $50 \times 25{,}000$ for CIFAR10. These matrices consist of the flattened activations at the layer before classification, i.e. the intermediate activations $\phi(x)$. We can thus obtain 50 correlation coefficients for the 50 neuron dimensions. We repeat this process at different training epochs, and compare the representation at a given epoch with the initial one. Note that all compared networks, without and with AL2, are initialized with the same set of random weights for a fair comparison. Since CCA is scale-invariant, this metric only depicts structural similarities between representations and can thus provide a good insight into the representation progress. A simple scaling down of the activation values does not affect the similarity measure. Figure 1 therefore shows that AL2 significantly modifies the learning and the final learned representations. As supported by the results reported in Section 4, including our regularization thus pushes the network's learning towards a fundamentally different representation, reducing the effect of overfitting.

5.2. Generalization analysis with cumulative ablations

Analyzing generalization can also be carried out with a different approach than randomization experiments. A recent approach shows the correlation between network generalization and the area under the cumulative ablation curve [12]. This cumulative ablation curve is defined by the authors as the accuracy of the pre-trained network for different percentages of ablations going from zero to 100%, on the training set. An ablation of 20% consists of systematically setting 20% of the activations in the feature representation layer to zero during the feed-forward inference of the pre-trained network. These ablations are also said by the authors to be related to sharpness [20], which is found to be a good indicator of generalization strength when combined with a norm metric [24].

We apply cumulative ablations on our pre-trained networks and report results in Table 2. At inference time, the activations of the feature representation layer obtained with the trunk $\phi(\cdot)$ of the network are set to zero at increasing rates going from zero to 100% in steps of 10. We measure the accuracy with each of the ablation rates, and calculate the area under the curve. We repeat this procedure at an interval of 100 epochs for each of the 8 networks to create the results of Table 2. The generalization assessment results are consistent with those discussed in Section 4, and confirm our previous observations (this is the case across our diverse experiments and datasets). We also note that even between epochs 300 and 400, where the dropout network still does not overfit and performs similarly without and with AL2 (Table 1), the area under the cumulative ablation curve is larger with AL2 training as shown in Table 2.

6. CONCLUSION

We propose a novel progressive activation loss (AL2) to regularize neural networks. Our loss acts increasingly with epochs on the magnitude of the activation signals of the feature representation layer. We use canonical correlation analysis to study the effect of AL2 on the learned feature representation throughout the training. This shows empirically the significant effect of our regularization on the fundamental representation that is learned by the networks.

We analyze memorization and generalization with randomization tests and with a cumulative ablation study to show the improvements of our AL2 method over state-of-the-art regularization techniques on three standard benchmark datasets.

7. REFERENCES

[1] Y. Xu, T. Mo, Q. Feng, P. Zhong, M. Lai, I. Eric, and C. Chang, "Deep learning of feature representation with multiple instance learning for medical image analysis," in *ICASSP*, 2014, pp. 1626–1630.

[2] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving semantic segmentation via video propagation and label relaxation," in *CVPR*, 2019, pp. 8856–8865.

[3] Y. Kim, H. Lee, and E. M. Provost, "Deep learning for robust feature generation in audiovisual emotion recognition," in *ICASSP*, 2013, pp. 3687–3691.

[4] L. Deng, D. Yu, and J. Platt, "Scalable stacking and learning for building deep architectures," in *ICASSP*, 2012, pp. 2133–2136.

[5] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *ICCV*, 2017, pp. 2051–2060.

[6] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, et al., "A closer look at memorization in deep networks," in *ICML*, 2017, pp. 233–242.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[9] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *NeurIPS*, 1992, pp. 950–957.

[10] M. Blot, T. Robert, N. Thome, and M. Cord, "Shade: Information-based regularization for deep learning," in *ICIP*, 2018, pp. 813–817.

[11] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *ICLR*, 2017.

[12] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization," in *ICLR*, 2018.

[13] X. Li, S. Chen, X. Hu, and J. Yang, "Understanding the disharmony between dropout and batch normalization by variance shift," in *CVPR*, 2019, pp. 2682–2690.

[14] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *NeurIPS*, 2018, pp. 8527–8537.

[15] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *CVPR*, 2018, pp. 9446–9454.

[16] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," in *NeurIPS*, 2017, pp. 6076–6085.

[17] A. Morcos, M. Raghu, and S. Bengio, "Insights on representational similarity in neural networks with canonical correlation," in *NeurIPS*, 2018, pp. 5727–5736.

[18] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro, "Towards understanding the role of over-parametrization in generalization of neural networks," in *ICLR*, 2019.

[19] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?," in *ICML*, 2019.

[20] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," in *ICLR*, 2017.

[21] E. Hoffer, I. Hubara, and D. Soudry, "Train longer, generalize better: closing the generalization gap in large batch training of neural networks," in *NeurIPS*, 2017, pp. 1731–1741.

[22] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio, "Sharp minima can generalize for deep nets," in *ICML*, 2017, pp. 1019–1028.

[23] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro, "The implicit bias of gradient descent on separable data," *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 2822–2878, 2018.

[24] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, "Exploring generalization in deep learning," in *NeurIPS*, 2017, pp. 5947–5956.

[25] P. L. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," *The Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2002.

[26] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. M. Erfani, S.-T. Xia, S. Wijewickrema, and J. Bailey, "Dimensionality-driven learning with noisy labels," in *ICML*, 2018, pp. 3361–3370.

[27] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?," in *ICML*, 2019, pp. 7164–7173.

[28] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels," in *CVPR*, 2019, pp. 7017–7025.

[29] M. El Helou, S. Mandt, A. Krause, and P. Beardsley, "Mobile robotic painting of texture," in *ICRA*, 2019.