NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

as a manuscript

Arsenii Ashukha

PRIOR KNOWLEDGE FOR DEEP LEARNING

PhD Dissertation Summary
for the purpose of obtaining academic degree
Doctor of Philosophy in Computer Science

Moscow — 2022

The PhD Dissertation was prepared at National Research University Higher School of Economics.

Academic Supervisor: Dmitry P. Vetrov, Candidate of Sciences, National Research University Higher School of Economics.

Contents

1 Introduction

Topic of the thesis

This work studies ways to introduce prior knowledge to deep learning models. The work proposes *sparse variational dropout*, a method that allows for a high level of sparsification in deep neural networks based on a sparsity-inducing prior distribution. To introduce a more expressive prior distribution that can account for correlations between weights and cover multiple modes, we propose a *deep weight prior* (DWP). DWP is a way of using complex generative models as a prior distribution over weights of a deep convolutional neural network.

Further, we focus on averaging predictions over objects' augmentations during inference, so-called test-time augmentation (TTA). TTA is a way to correct an imperfectly acquired prior knowledge of a model to improve its predictive performance. We propose TTA for ensembles, an algorithm for learning a data-augmentation policy for the TTA, and a way to improve the quality of representations based on TTA.

Actuality of the work

Machine learning is a field of science which focuses on the creation of data-based predictive models. The main distinctive feature of machine learning is its ability to make predictions when simple rule-based approaches fail. Machine learning automatically reconstructs predictive rules and representations, even if it requires finding complex dependencies in data.

The most successful family of machine learning models for complexly structured data such as images, videos, natural language, sound or molecules is neural networks [22]. Roughly speaking, deep neural networks (DNNs) can be defined as a composition of parameterized differentiable modules. The success of deep neural networks is heavily based on:

i) large data collections that include from millions to hundreds of millions of data points;

ii) highly adjustable transformations, that can include hundreds of billions of parameters;

ii) a lot of computations that are required to adjust large models on large data collections.

Acquiring large data collections or using huge models is not always possible. Huge models require a lot of energy and thus cannot run on low-power devices. The collection and labeling of a dataset are always slow and costly. To overcome this issue, one can use models that leverage prior knowledge.

Prior knowledge—comes from the Latin phrase *a priori* («from what is before»)—refers to information, independent of a dataset under consideration. The prior information can be utilized in a machine learning algorithm in order to improve its quality, computational budget, training speed, etc. Prior knowledge can be integrated into deep learning models via the following components:

i) **The design of architectures.** The design of deep learning architectures often implies the usage of prior information. The most notable examples are modern convolutional neural networks [18] that use information on the spatial structure of data, and equivariant convolutional networks [3] where—motivated by needs of certain domains—an output of a model changes predicatively and smoothly under specific groups of natural transformations of an input.

ii) **Data augmentation** Data augmentation is a popular tool that artificially expands a number of objects in training data. The design of data augmentation strategy exploits predefined input transformations. These transformations are usually label-preserving; otherwise, they change a label in a predictable manner. The design of these transformations is often built upon prior knowledge on a specific problem.

iii) **Prior distribution over parameters.** Bayesian approach is another popular way to integrate prior knowledge into deep learning models. It assumes that prior knowledge is available as a distribution over unobserved variables ν. Bayesian approach allows to access an approximation of a posterior distribution or simply an approximate posterior over unobserved variables after observing a dataset, a likelihood $p(\text{data} \mid \nu)$, and a prior distribution.

$$\underbrace{p(\nu \mid \text{data})}_{\substack{\text{intractable posterior distribution}}} = \frac{p(\text{data} \mid \nu) p_{\text{prior}}(\nu)}{\int_\nu p(\text{data} \mid \nu) p_{\text{prior}}(\nu)} \approx \underbrace{q_\phi(\nu)}_{\substack{\text{approximate posterior} \\ \text{result of variational inference}}} \tag{1}$$

In case of deep learning models, Bayesian approach is used to infer an approximate posterior over model weights. Prior distribution allows to specify preferences about weights that will be accounted during inference of the approximate posterior.

In principle, if a dataset is large enough, using the prior information might not be necessary. For example, an architecture called vision transformer [6] or MLP-Mixer [31] can give competitive or even state-of-the-art performance on computer vision benchmarks without explicitly using convolutions or other manually defined local transformations. These architectures usually require more data and more parameters compared to prior-rich models, but can avoid the pitfalls of manually designed—thus not always optimal—priors that are built in prior-rich architectures.

Humans, just like deep neural networks, are heavily exploiting priors. Brain architecture is a result of many years of evolution. Human brain is designed to make humans learn fast, avoid danger, and process a ton of various signals every second. Humans use experiences gained through life to solve a new problem but may struggle when there is little prior information available or the knowledge transfer system fails [7]. This makes us speculate that prior information is needed for artificial neural networks and will be an essential component for artificial intelligence to emerge.

The goal of this work is to develop mechanisms of prior knowledge integration and use these mechanisms to improve deep learning models.

2 Key results and conclusions

The novelty of this work can be summarized as follows:

1. We proposed *sparse variational dropout*, a method that, based on sparsity inducing prior, allows for sparsification of deep neural networks. The method for the first time showed an ability of variational dropout [16] to learn sparse models. To make the training of *sparse variational dropout* possible we proposed to use a noise reduction parametrization, and the local reparametrization trick for convolutions.
2. We proposed *deep weight prior*, a method that allows to use a variational auto-encoder [17]—an expressive generative model—as a prior distribution over weights of a deep learning model. To that end, we proposed a variational lower bound that allows to use implicit prior distributions for variational inference and log-likelihood training.
3. We propose *test-time augmentation (TTA) for ensembles*, a simple yet well-performing method, that allows to integrate prior knowledge to an ensemble of deep learning models. TTA improves predictive performance of ensembles [19; 21] with a negligible additional computational cost.
4. We proposed *greedy policy search*, a method that can learn a data augmentation policy for test-time augmentation (TTA). In other words, the method can automatically select priors that need to be used during TTA. That helps to avoid too aggressive augmentations that are harmful to the predictive performance.
5. We proposed *mean embeddings with test-time data augmentation*, a method that adapts test-time augmentation for inference of embeddings, integrating prior knowledge not only to predictions but also to representations.

Theoretical and practical significance. We propose a sparsification method that can be used for compression and acceleration of deep learning models. Compression and acceleration are important for running neural networks on low-powered devices. Another outcome of this work is a framework that allows to use flexible prior distributions, expressing complex beliefs about the favorable weight distributions. For example, a prior can represent correlations between the weights and cover multiple modes. We have also developed methods for using and learning policies for ensembling over augmentations during inference. Usage of these policies increases the model robustness, which is essential for safety-critical applications, e.g. medical diagnosis.

Methodology and research methods. In this work, we apply deep learning, doubly stochastic variational inference, probabilistic modeling, generative models, data augmentation, as well as continuous and discrete optimizations.

Reliability We provide a detailed description of proposed methods and experiments. For all papers, code was made publicly available.

Key aspects/ideas to be defended.

1. A finding that *variational dropout* learns sparse solutions.
2. *Deep weight prior*, a framework that allows to use implicit distributions for variational inference and log-likelihood training.
3. *Test-time augmentation* for ensembles.
4. *Greedy policy search*, an algorithm for learning test-time data augmentation policies.
5. *MeTTA*, an algorithm for using ensembling for improving representations.

Personal contribution. The author of this dissertation obtained all the stated results. In all cases mentioned, both text and experimental results presented in a paper are the result of collaboration between all authors. In the first paper «Variational dropout sparsifies deep neural networks», the author proposed to use local reparametrization trick for convolutional layers, contributed to training algorithm, and discovered sparsification effect on deep convolutional networks. In «The Deep Weight Prior», the author proposed a core idea and derived a model. In «Pitfalls of In-Domain Uncertainty Estimation and Ensembling in Deep Learning», the author proposed to use test-time augmentation (TTA) for ensembles. In «Mean Embeddings with Test-Time Data Augmentation for Ensembling of Representations», the author proposed to use TTA to enhance the quality of representations. In «Greedy Policy Search: A Simple Baseline for Learnable Test-Time Augmentation», the author contributed to the core idea and performed in-domain ImageNet experiments.

Publications and probation of the work

* denotes equal contribution of coauthors

First-tier publications

1. *Dmitry Molchanov*, Arsenii Ashukha*, and Dmitry Vetrov* Variational dropout sparsifies deep neural networks // In International Conference on Machine Learning, pp. 2498-2507. PMLR, 2017. CORE A* conference.

2. *Alexander Lyzhov*, Yuliya Molchanova*, Arsenii Ashukha*, Dmitry Molchanov*, Dmitry Vetrov* Greedy policy search: a simple baseline for learnable test-time augmentation // In Conference on Uncertainty in Artificial Intelligence, 2020. CORE A* conference.

Second-tier publications

1. *Andrei Atanov*, Arsenii Ashukha*, Kirill Struminsky, Dmitry Vetrov, Max Welling* The deep weight prior // International Conference on Learning Representations, ICLR 2019. Indexed by SCOPUS. From 2021 ICLR is CORE A* conference.

Other publications

1. *Arsenii Ashukha**, *Alexander Lyzhov**, *Dmitry Molchanov**, *Dmitry Vetrov* Pitfalls of in-Domain uncertainty estimation and ensembling in deep learning // International Conference on Learning Representations, ICLR 2020. From 2021 ICLR is CORE A* conference.

2. *Arsenii Ashukha, Andrei Atanov, Dmitry Vetrov* Mean embeddings with test-time data augmentation for ensembling of representations // Uncertainty & Robustness in Deep Learning, ICML, 2021.

Reports at conferences and seminars.

1. Poster presentation on «Variational dropout sparsifies deep neural networks», International Conference on Machine Learning, Sidney, 2017.
2. Talk on «Variational dropout sparsifies deep neural networks», Seminar of Bayesian methods research group, Moscow, 2017.
3. Poster presentation on «The deep weight prior», International Conference on Learning Representations, New Orleans, USA, 2019
4. Poster presentation on «Greedy policy search: a simple baseline for learnable test-time augmentation», Uncertainty in Artificial Intelligence, virtual, 2020.

Volume and structure of the work. The thesis contains an introduction, contents of publications and a conclusion. The full volume of the thesis is 107 pages.

The author has also contributed to the following publications

1. *Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, Dmitry Vetrov* Structured Bayesian pruning via log-normal multiplicative noise // International Conference on Neural Information Processing System, NeurIPS 2017. Core A* conference.

2. *Kirill Neklyudov**, *Dmitry Molchanov**, *Arsenii Ashukha**, *Dmitry Vetrov* Variance networks: when expectation does not meet your expectations // International Conference on Learning Representations, ICLR 2019. Indexed by SCOPUS. From 2021 ICLR is CORE A* conference.

3. *Andrei Atanov, Arsenii Ashukha, Dmitry Molchanov, Kirill Neklyudov, Dmitry Vetrov* Uncertainty estimation via stochastic batch normalization // ICLR Workshop Track 2018 // In International Symposium on Neural Networks, pp. 261-269. Springer, Cham, 2019.

4. *Max Kochurov, Timur Garipov, Dmitry Podoprikhin, Dmitry Molchanov, Arsenii Ashukha, Dmitry Vetrov* Bayesian incremental learning for DNNs // ICLR Workshop Track 2018.

5. *Evgenii Nikishin, Arsenii Ashukha, Dmitry Vetrov* Unsupervised domain adaptation with shared latent dynamics for reinforcement learning // Bayesian DL, NIPS 2019.

6. *Andrei Atanov, Alexandra Volokhova, Arsenii Ashukha, Ivan Sosnovik, Dmitry Vetrov* Semi-conditional normalizing flows for semi-supervised learning // Workshop on Invertible Neural Nets and Normalizing Flows, ICML, 2019.

3 Content of the work

3.1 Variational dropout sparsifies deep neural networks

Deep neural networks *de facto* have become a tool of choice for many real-world machine learning problems from detection [34] and translation [33] all the way to protein-folding [15]. However, excellent performance always comes at the cost of a large number of parameters, which leads to high memory and computational requirements. One way to mitigate this issue is to learn a sparse model, where most of the weights are equal to zero. Thus, model size and computational cost will be reduced. In the work «Variational Dropout Sparsifies Deep Neural Networks» we propose a model called *sparse variational dropout* that can train highly sparse deep neural networks.

Model description

We consider a supervised learning problem with a dataset $D = (x_i, y_i)_{i=0}^N$, where x_i is an object and y_i is its label. The model is trained via variational inference over weights of deep neural networks. The goal of variational inference is to learn a variational approximation $q_\phi(W) \approx p(W \mid D)$, where ϕ are the parameters of variational approximation to be trained. The objective of variational inference is a stochastic estimate of a variational lower bound $\mathcal{L}(\phi)$:

$$\mathcal{L}(\phi) \simeq \mathcal{L}^{SGVB}(\phi) = L_{\mathcal{D}}^{SGVB}(\phi) - D_{KL}(q_\phi \| p) \to \max_\phi \tag{2}$$

$$L_{\mathcal{D}}(\phi) \simeq L_{\mathcal{D}}^{SGVB}(\phi) = \frac{N}{M} \sum_{m=1}^{M} \log p(y_m | x_m, \hat{W}), \quad \hat{W} \sim q(W \mid \phi) \tag{3}$$

where $L_{\mathcal{D}}^{SGVB}$ is an estimate of a log-likelihood that is defined via a deep neural network $p(y_m|x_m, \hat{W})$, and $p(W)$ is a prior distribution over weights of the deep neural network, $D_{KL}(u, v) = \int dx\, u(x) \log \frac{u(x)}{v(x)}$ is KullbackLeibler divergence, also called relative entropy which is an asymmetric distance between two distributions. $L_{\mathcal{D}}^{SGVB}$ plays a role of a data-term that controls how well the model $p(y_m|x_m, \hat{W})$ performs on the training set, and $D_{KL}(q_\phi, p)$ controls how well the model satisfies the prior distribution $p(W)$. We also assume that a $q_\phi(W)$ supports reparameterization [17; 29].

In sparse variational dropout, we use fully-factorized Gaussian variational approximation in the additive parametrization:

$$q(w_{ij} \mid \theta_{ij}, \alpha\theta_{ij}) = \underbrace{\mathcal{N}(w_{ij} \mid \theta_{ij}, \alpha_{ij}\theta_{ij}^2)}_{\substack{\text{multiplicative parametrization} \\ \textbf{(author?)}\ [16]}} = \underbrace{\mathcal{N}(w_{ij} \mid \theta_{ij}, \mu_{ij}^2)}_{\substack{\text{additive parametrization} \\ \text{proposed in this work}}}, \tag{4}$$

that has an noise reduction propriety for the gradients over parameters θ_{ij}, and the log-uniform prior distribution

$$p(\log|w_{ij}|) = \text{const} \quad \Leftrightarrow \quad p(|w_{ij}|) \propto \frac{1}{|w_{ij}|}. \tag{5}$$

The algorithm uses *local reparametrization trick* (LRT) [16] for both fully-connected and convolutional layers. LRT computes a distribution over activations on each layer analytically, in an assumption that an input is not a random variable. As a result, it allows to reduce the variance of a gradient. LRT can be interpreted it a layer-wise per-objects samples of weights $\hat{W}$ from an approximate posterior $q(W \mid \phi)$.

Empirical evaluation

Sparse variational dropout has been tested on MNIST [23], CIFAR-10, and CIFAR-100 [18] classification benchmarks. On MNIST, a method provides up to $280\times$ fewer parameters than the original dense network. The results are presented in Table 1. On CIFAR datasets, the sparsification rate increases as a width of a model increases, while error rates are closely matched with dense models.

Also, we show that sparse variational dropout removes 100% of weights of a model when there is no connection between objects and labels which is so-called random labeling [35] setting.

Network	Method	Error %	Sparsity per Layer %	$\frac{\lvert W\rvert}{\lvert W_{\neq 0}\rvert}$
	Original	1.64		1
	Pruning	1.59	$92.0 - 91.0 - 74.0$	12
LeNet-300-100	DNS	1.99	$98.2 - 98.2 - 94.5$	56
	SWS	1.94		23
(ours)	Sparse VD	1.92	$98.9 - 97.2 - 62.0$	**68**
	Original	0.80		1
	Pruning	0.77	$34 - 88 - 92.0 - 81$	12
LeNet-5-Caffe	DNS	0.91	$86 - 97 - 99.3 - 96$	111
	SWS	0.97		200
(ours)	Sparse VD	0.75	$67 - 98 - 99.8 - 95$	**280**

Table 1: Comparison of various sparsification techniques (Pruning [13; 12], DNS [11], SWS [32]) on LeNet-300-100 (3 layers) and LeNet-5-Caffe (4 layers) architectures. Sparse variational dropout provides the highest level of sparsity at the same level of accuracy.

Retrospective

Sparse variational dropout works in practice, and it was used for network sparsification in leading IT companies. However, future studies showed that careful usage of pruning-based methods can produce better results [10]. Sparse solution is just a local optimum, as better values of ELBO can be achieved with a less flexible variational posterior $q(w_{ij}) = N(w_{ij} \mid 0, \sigma_{ij})$ [27].

Training DNNs with noise is known to be a hard and unstable process. That is less the case with SparseVD. All variances are initialized with small values and do not change much during training. The usage of small variances neither hurts the performance nor introduces a lot of noise. Thus, SparseVD might be considered as a low-noise regularizer.

3.2 The deep weight prior

Variational inference is a tool that, after observing data, allows to transform a prior distribution over parameters of a machine learning model to an approximate posterior distribution. Priors played an important role in many recent models for quantization [32], sparsification [26; 28], and compression [25]. Despite that, they are limited to a fully factorized setting and cannot cover multiple modes. In this work, we propose *deep weight prior*, a framework that allows to train and use complex prior distributions that are defined via an implicit generative model:

$$\hat{p}_l(w) = \int p(w \mid f_\phi(z)) p_l(z)\, dz, \tag{6}$$

where a conditional distribution $p(w \mid f_\phi(z))$ is an explicit parametric distribution, f_ϕ is a neural network, and $p_l(z)$ is an explicit prior distribution that does not depend on trainable parameters. The distribution in equation (6) can be interpreted as a mixture of infinitely many distributions. Implicit priors can capture complex distributions over weights that can better express prior beliefs.

Training a prior distribution

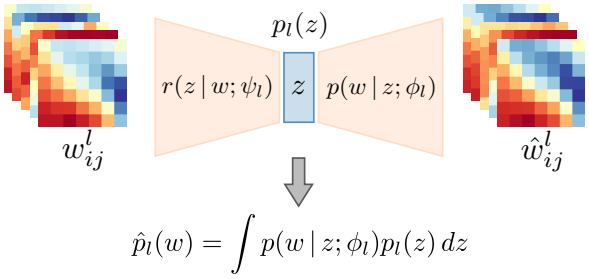
In order to train a prior distribution in the form of eq. 6, we do the following steps:

1. define a specific architecture of a prior in the form of a variational auto-encoder;
2. collect a set of networks trained on an available dataset;
3. train the prior distribution on the weights of the trained networks.

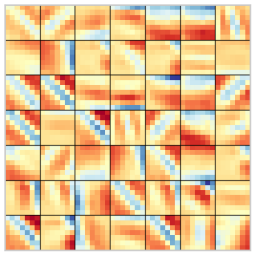
The trained prior distribution can be used for variational inference on a new dataset. Fig. 1a shows how the prior can be constructed from the trained variational auto-encoder model. Figs. 1b, 1c demonstrate samples from the trained generative models compared to the real kernels.

Variational inference with implicit priors

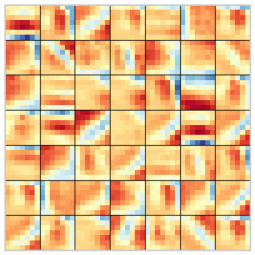
We consider a supervised learning problem with a dataset $D = (x_i, y_i)_{i=0}^N$, where x_i is an object and y_i is its label. We train a model $p(y_i \mid x_i, W)$ with variational inference that optimizes the



(a) Learning of DWP with VAE



(b) Learned filters



(c) Samples DWP

Figure 1: In subfig. 1a we show the process of learning a prior distribution over kernels of one convolutional layer. First, we train encoder $r(z \mid w; \phi_l)$ and decoder $p(w \mid z; \psi_l)$ with VAE framework [17]. Then, we use the decoder to construct the prior $\hat{p}_l(w)$. In subfig. 1b we show a batch of learned kernels of shape 7×7 from the first convolutional layer of a CNN trained on NotMNIST dataset. In subfig. 1c we show samples from the deep weight prior that is learned on trained kernels.

following objective:

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \mathbb{E}_{q_\theta(W)} \log p(y_i \mid x_i, W) - D_{\mathrm{KL}}(q_\theta(W) \| p(W)) \to \max_\theta, \tag{7}$$

where W denotes weights of a neural network, $q_\theta(W)$ is a variational distribution that allows reparametrization [17; 9], and $p(W)$ is a prior distribution.

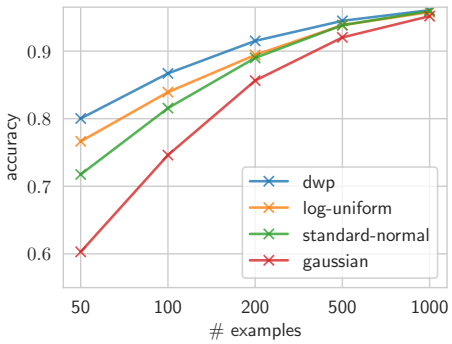
We consider a neural network with L convolutional layers and denote parameters of l-th convolutional layer as $w^l \in \mathrm{R}^{I_l \times O_l \times H_l \times W_l}$, where I_l is the number of input channels, O_l is the number of output channels, H_l and W_l are spatial dimensions of kernels. The parameters of the neural network are denoted as $W = (w^1, \dots w^L)$. A variational approximation $q_\theta(W)$ and a prior distribution $p(W)$ have the following factorization over layers, filters, and channels:

$$q_\theta(W) = \prod_{l=1}^{L} \prod_{i=1}^{I_l} \prod_{j=1}^{O_l} q(w_{ij}^l \mid \theta_{ij}^l) \qquad p(W) = \prod_{l=1}^{L} \prod_{i=1}^{I_l} \prod_{j=1}^{O_l} p_l(w_{ij}^l), \tag{8}$$

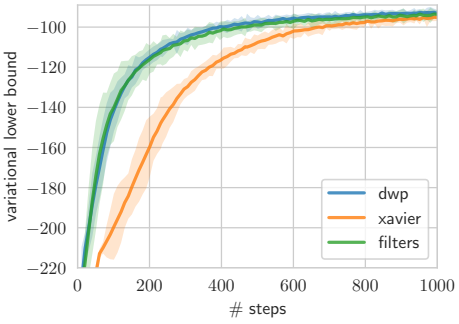
where $w_{ij}^l \in \mathrm{R}^{H_l \times W_l}$ is a kernel of j-th channel in i-th filter of l-th convolutional layer.

KL-divergence with implicit priors (eq. 6) cannot be computed in a closed-form or unbiasedly estimated. To make the computation of the variational lower bound tractable, we introduce an auxiliary lower bound on the KL-divergence:

$$D_{\mathrm{KL}}(q(W) \| \hat{p}(W)) = \sum_{l,i,j} D_{\mathrm{KL}}(q(w_{ij}^l | \theta_{ij}^l) \| \hat{p}_l(w_{ij}^l)) \le \sum_{l,i,j} \big(- H(q(w_{ij}^l \mid \theta_{ij}^l)) +$$
$$+ \mathbb{E}_{q(w_{ij}^l \mid \theta_{ij}^l)} \big[D_{\mathrm{KL}}(r(z \mid w_{ij}^l; \psi_l) \| p_l(z)) - \mathbb{E}_{r(z \mid w_{ij}^l; \psi_l)} \log p(w_{ij}^l \mid z; \phi_l) \big] \big) = D_{\mathrm{KL}}^{bound}, \tag{9}$$



(a) ConvNet on MNIST



(b) VAE on MNIST

Figure 2: (2a) For training subsets of different size, we demonstrate the performance of variational inference with a fully-factorized variational approximation with different prior distributions: *deep weight prior* (dwp), log-uniform, standard normal, trained normal with full covariance matrix (gaussian). We have found that variational inference with a deep weight prior distribution achieves better mean test accuracy than learning with other priors. (2b) Initialization of weights of the models with deep weight priors or learned filters significantly increases the training speed, compared to Xavier initialization. This provides a shred of evidence that *deep weight prior* closely matches the true distribution of kernels.

where $r(z \mid w; \psi_l)$ is an auxiliary inference model for the prior of l-th layer $\hat{p}_l(w)$, The final auxiliary variational lower bound $\mathcal{L}^{aux}(\theta, \psi)$ has the following form:

$$\mathcal{L}^{aux}(\theta, \psi) = L_D - D_{\mathrm{KL}}^{bound} \leq \mathcal{L}(\theta) = L_D - D_{\mathrm{KL}}(q_\theta(W) \| \hat{p}(W)). \tag{10}$$

Empirical results

We demonstrated that using *deep weight prior* can be beneficial when training with limited labeled data. An initialization of weights with *deep weight prior* allows to converge faster. The results are shown in Fig. 2.

Retrospective

In general, *deep weight prior* is hard to train for models and datasets beyond MNIST size. However, it is possible that we do not have the knowledge on what kind of data DWP works the best. For example, [20] successfully applied *deep weight prior* to 3D magnetic resonance imaging (MRI).

Generative models that generate weights of MLP-based representations (hypernetworks) have recently become popular [30; 8; 2]. *Deep weight prior* can be considered as a small step toward generative models over trained neural networks that might be an important topic for future research on generative models.

3.3 Test-time data augmentation improves ensembles for free

Test-time augmentation (TTA)[18] is a simple technique that averages the prediction of a model w.r.t. different augmentations. TTA makes predictions invariant to the augmentation transformations and allows to improve predictive performance of a deep learning model.

We propose to use TTA for ensembles. In this case, each member of an ensemble is applied to a different sample of an augmented object (equations 12, 13).

$$p(y \mid x) = \frac{1}{S} \sum_{i=1}^{S} p(y|x, \theta_i) \qquad (11)$$

(a) Classical ensembling

$$p(y \mid x) = \frac{1}{S} \sum_{i=1}^{S} p(y \mid \hat{x}_i, \theta_i), \qquad (12)$$

$$\text{where } \hat{x}_i \sim p_{\text{aug}}(\cdot \mid x) \qquad (13)$$

(b) Ensembling with test-time augmentation

TTA improves ensembles with a negligible additional computational cost during inference. Empirical performance of the method is demonstrated via various ResNet50 [14] ensembles on ImageNet (Fig. 4). Interestingly, a single model with TTA performs competitively with methods that require significantly larger number of parameters, computational budget, and training complexity. TTA is a simple yet powerful approach that was overlooked by uncertainty estimation community.

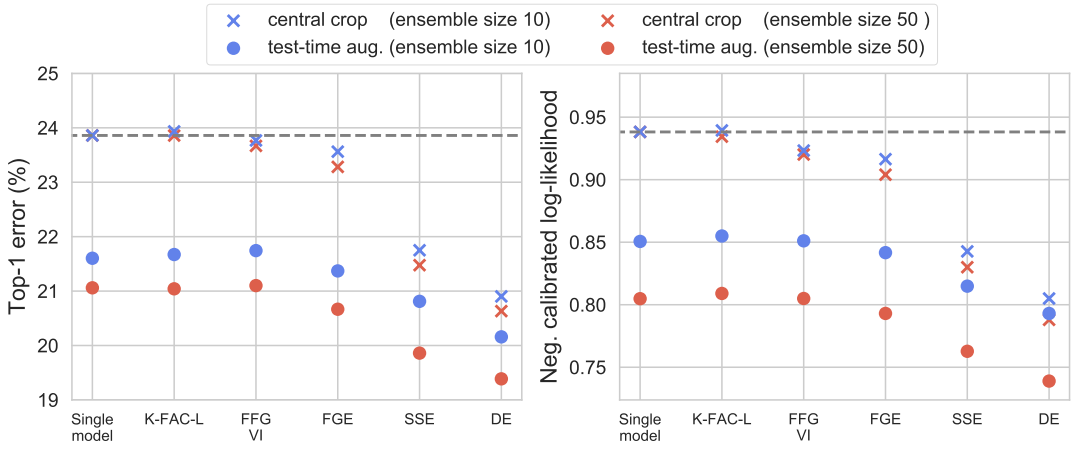


Figure 4: How to read results: $\times \xrightarrow[10 \text{ samples}]{\text{test-time aug}} \bullet$, $\times \xrightarrow[50 \text{ samples}]{\text{test-time aug}} \bullet$. The negative calibrated log-likelihood (lower is better) for different ensembling techniques on ImageNet. We report performance for two regimes. *Central-crop evaluation* ($\times\times$) means every member of an ensemble is applied to a central crop of an image, and *test-time data augmentation* ($\bullet$ $\bullet$) means each member of the ensemble is applied to a separate random augmentation of the image. **Test-time data augmentation significantly improves ensembles with no additional computational cost.**

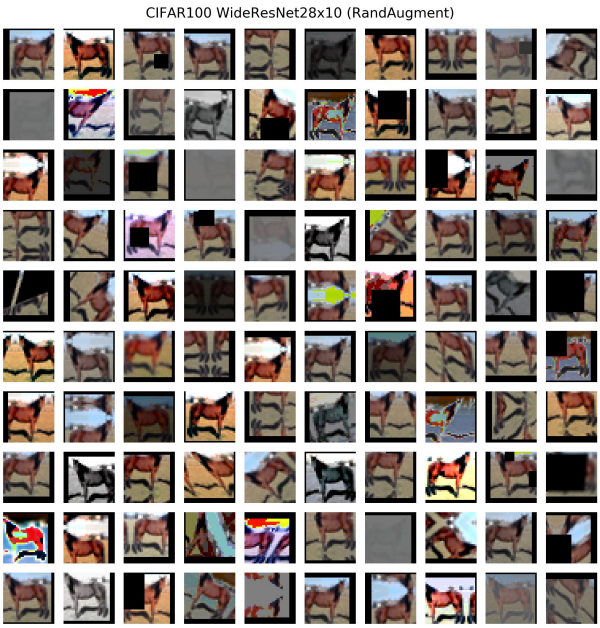


Figure 5: An illustration of a test-time augmentation policy trained with greedy policy search.

3.4 Greedy policy search for learnable test-time augmentation

Data augmentation is one of the popular techniques for training deep neural networks. It allows to expand a dataset size and pass prior knowledge of domain invariances to a deep neural network. However, in most cases, the trained deep neural networks are not fully invariant to data augmentation transforms. In other words, the DNNs do not acquire prior knowledge entirely correctly. To alleviate that issue, test-time data augmentation is used. It accounts for several samples of augmentations for every object during inference. However, modern data augmentation techniques that work the best on a training stage usually do not perform best during inference. One reason for it is that data augmentation with a high level of noise, e.g. RandAugment [4], can act as regularization during training but might degrade the performance during inference-time averaging.

In this work, we want to use prior knowledge on a set of label-preserving transformations in order to learn an optimal policy for test-time data augmentations. The primal goal is to demonstrate that learning of a test-time augmentation policy is possible.

Greedy Policy Search

Greedy policy search (*GPS*) iteratively builds a test-time data augmentation policy. We will consider the problem of learning a policy for a single pre-trained classification model. During the inference stage, the predictions of a model will be averaged w.r.t augmentations of an object that are obtained with the trained policy. The illustration of samples from the trained GPS-policy is available in Fig. 5.

GPS is organized as follows:

i) It starts with sampling a large pool of sub-policies. Each sub-policy is a combination of two-three randomly selected transformations with randomly sampled magnitudes.

ii) GPS selects a sub-policy that improves the current policy the most, and adds it to the current policy.

iii) Repeats the step *ii* while a number of sub-policies is less than required.

We find that conventional accuracy as a search criterion works significantly worse than calibrated negative log-likelihood. Accuracy is likely too noisy, and predictions require calibration in order to alleviate possibly wrong temperature. Despite being simple, *greedy policy search* works better than more complex RL-based optimization [24].

Empirical results

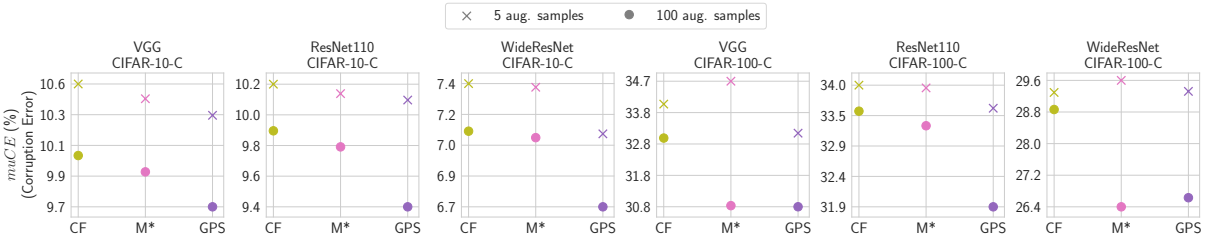


Figure 6: Mean unnormalized corruption error ($muCE$) on corrupted versions of CIFAR datasets for various test-time augmentation strategies: random crops and horizontal flips (CF), modified RandAugment with M found by grid search (M^*) and GPS policy (GPS). Learnable TTA methods were trained on clean, uncorrupted data. In most cases, GPS policies are more robust to the domain shift compared to alternatives.

Empirically, we demonstrate that *greedy policy search* improves the results of classical test-time augmentation on in- and out-of-domain data, without any additional computational cost during inference. GPS can also be applied to an ensemble. A sample of results is shown in Figure 6.

3.5 Mean embeddings for ensembling of representations

Ensembling is a popular tool that improves uncertainty estimates and the pure quality of deep learning models. However, ensembles are usually applied to improving the prediction of a model and cannot be applied to improving the quality of representations, e.g. independent networks have misaligned representations. However, improving the quality of representations is important for many problems, e.g. image retrieval, content matching, verification, and recommendation systems.

We propose ***mean embeddings with TTA*** (***MeTTA***)—a simple method for representations ensembling. The method averages representations from the L-th layer of a model $a^L(\cdot\,; \mathrm{w})$ of a

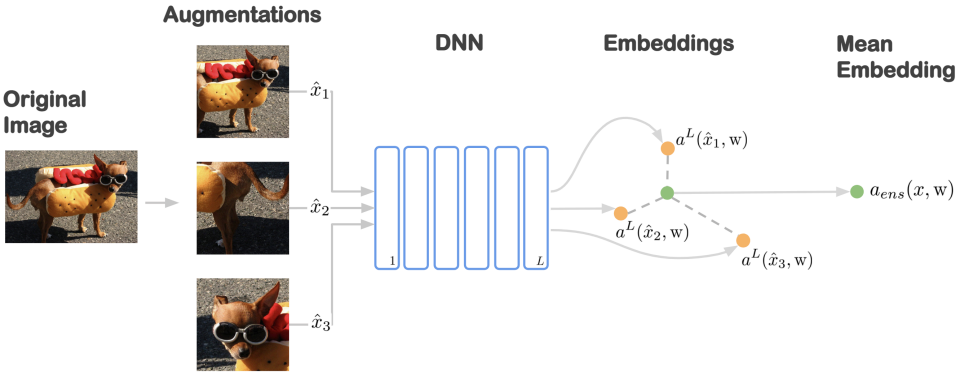


Figure 7: To produce a *mean embedding*, MeTTA averages activations of a network across different augmentations of an object. MeTTA does not affect the training phase and can be applied to a pre-trained network.

Problem	Model	Width	SK	# Params (M)	Central crop Embeddings	Mean embeddings with TTA $N = 10$	Mean embeddings with TTA $N = 32$
Self-supervised features (SimCLRv2)	ResNet50	1×	False	24	71.7	73.3 (+1.6%)	73.8 (+2.1%)
		1×	True	35	74.6	75.8 (+1.2%)	76.2 (+1.6%)
	ResNet101	2×	False	170	77.0	78.1 (+1.1%)	78.5 (+1.5%)
		2×	True	257	79.0	79.8 (+0.8%)	79.9 (+0.9%)
	ResNet152	3×	True	795	79.8	80.3 (+0.4%)	80.7 (+0.9%)
Supervised features	ResNet50	1×	False	24	76.6	78.0 (+1.4%)	78.5 (+1.9%)
		1×	True	35	78.5	79.7 (+1.2%)	80.2 (+1.7%)
	ResNet101	2×	False	170	78.9	80.2 (+1.3%)	80.6 (+1.7%)
		2×	True	257	80.1	81.0 (+0.9%)	81.3 (+1.3%)
	ResNet152	3×	True	795	80.5	81.4 (+0.9%)	81.9 (+1.4%)

Table 2: The comparison of different methods of inference embeddings. The table represents top-1 accuracy on ImageNet [5] for linear evaluation of embeddings with 100% labels. For self-supervised features, we used models that were pre-trained with SimCLRv2 [1]. We used both supervised and self-supervised pre-trained models from `https://github.com/google-research/simclr` repository. SK stands for selective kernels.

single model over different augmentations of an object x

$$a_{ens}(x; \mathrm{w}) = \mathbb{E}_{\hat{x} \sim p_{aug}(\cdot \,|\, x)} a^L(\hat{x}; \mathrm{w}) \cong \frac{1}{S} \sum_{s=1}^{S} a^L(\hat{x}_s; \mathrm{w}), \text{ where } \hat{x}_s \sim p_{aug}(\cdot \,|\, x), \tag{14}$$

where S is a number of samples of augmentations for a single image, w are the weighs of the network, and $a_{ens}(x; \mathrm{w})$ is a mean embedding. Empirically, the method works on both supervised and self-supervised models (Table 2). The illustration of MeTTA is available at Figure 7. The interpolation between the central crop embeddings and MeTTA embeddings are shown in Figure 8. MeTTA shows how ensembles can be applied to enhance the quality of representations. That has the potential to open many new applications.

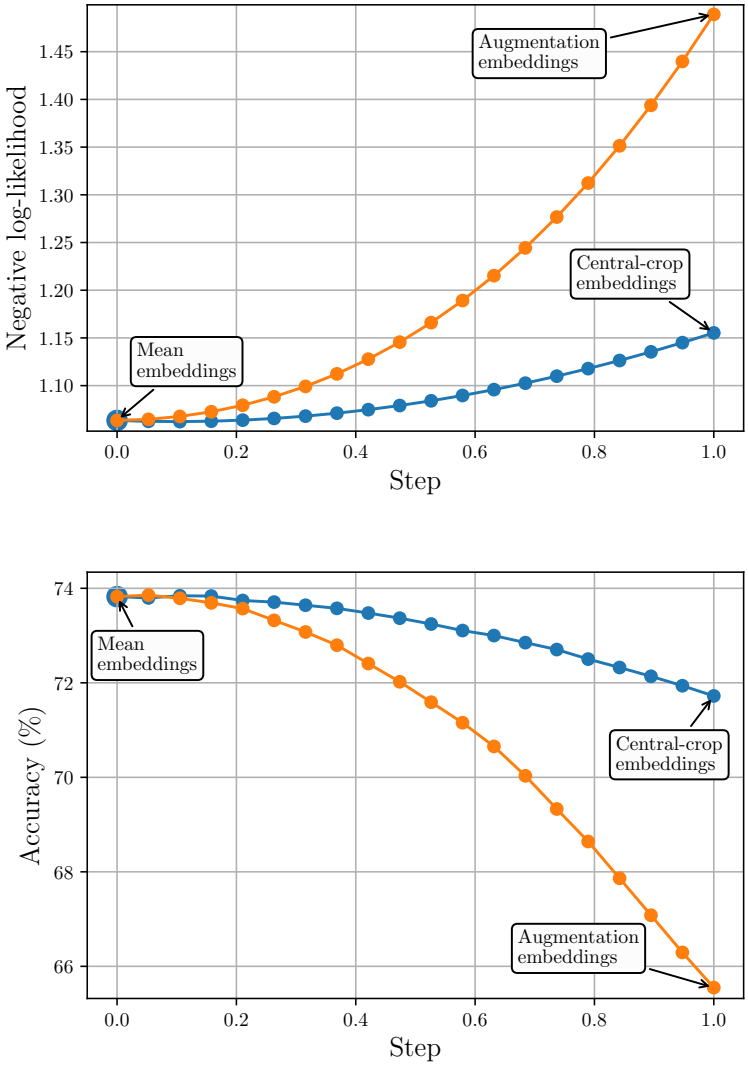


Figure 8: Negative log-likelihood (top) and accuracy (bottom) for linearly interpolated embeddings of the form $(1-\alpha)\cdot x+\alpha\cdot y$, where x is the mean embedding, and y is the central-crop embedding for blue and an embedding of an individual augmentation for orange. Metrics for each step size α are averaged over validation images for both curves and additionally over different augmentations for the orange one.

4 Conclusion

In the final section, we summarize the main contribution of the work.

1. We have proposed *sparse variational dropout*, a method for sparcification of deep neural networks. The method uses variational inference with the log-uniform prior distribution. To make training of *sparse variational dropout* possible, it has been proposed to use additive parametrization and local reparametrization trick that both reduce the variance of gradients. The method allows to learn models with a high sparsity level up to $270\times$ in our experiments.

2. We have proposed *deep weight prior*, a method that allows to train an expressive generative model over kernels of convolutional neural networks and use the generative model as a prior distribution during training of convolutional neural networks. To train *deep weight prior*, we developed a special form of variational inference that can work with an implicit prior distribution. Training with *deep weight prior* improves the quality of training with limited data and allows to converge faster.

3. We have proposed *test-time augmentation* for ensembles, which allows to diversify predictions and increase quality and uncertainty estimation ability of ensembles.

4. We have proposed *greedy policy search*, a method that allows to train a test-time data augmentation policy. Averaging predictions w.r.t. samples from the trained policy improves the performance of a network on in-domain and domain shift scenarios.

5. We have proposed MeTTA, a method that uses ensembling to enhance the quality of representations.

References

[1] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.

[2] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8628–8638, 2021.

[3] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[4] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Rachit Dubey, Pulkit Agrawal, Deepak Pathak, Thomas L Griffiths, and Alexei A Efros. Investigating human priors for playing video games. *arXiv preprint arXiv:1802.10217*, 2018.

[8] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. *arXiv preprint arXiv:2102.04776*, 2021.

[9] Michael Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. *arXiv preprint arXiv:1805.08498*, 2018.

[10] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[11] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pages 1379–1387, 2016.

[12] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[13] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, pages 1135–1143, 2015.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[15] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Kathryn Tunyasuvunakool, Olaf Ronneberger, Russ Bates, Augustin Žídek, Alex Bridgland, et al. High accuracy protein structure prediction using deep learning. *Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book)*, 22:24, 2020.

[16] Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Anna Kuzina, Evgenii Egorov, and Evgeny Burnaev. Bayesian generative models for knowledge transfer in mri semantic segmentation problems. *Frontiers in neuroscience*, 13:844, 2019.

[21] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

[22] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[24] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[25] Christos Louizos, Karen Ullrich, and Max Welling. Bayesian compression for deep learning. *arXiv preprint arXiv:1705.08665*, 2017.

[26] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. *arXiv preprint arXiv:1701.05369*, 2017.

[27] Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variance networks: When expectation does not meet your expectations. In *International Conference on Learning Representations*, 2019.

[28] Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry P Vetrov. Structured bayesian pruning via log-normal multiplicative noise. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[29] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 2014.

[30] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[31] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021.

[32] Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. *arXiv preprint arXiv:1702.04008*, 2017.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[34] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[35] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.