Learning from Small Data Through Sampling an Implicit Conditional Generative Latent Optimization Model

By

IDAN AZURI

Under the supervision of

PROF. DAPHNA WEINSHALL

THE HEBREW
UNIVERSITY
OF JERUSALEM

Faculty of Computer Science and Engineering
THE HEBREW UNIVERSITY OF JERUSALEM

A dissertation submitted to the Hebrew University of Jerusalem as a partial fulfillment of the requirements of the degree of MASTER OF SCIENCE in the Faculty of Computer Science and Engineering.

APRIL 2020

ABSTRACT

We revisit the long-standing problem of *learning from small sample*. In recent years major efforts have been invested into the generation of new samples from a small set of training data points. Some use classical transformations, others synthesize new examples. Our approach belongs to the second one. We propose a new model based on conditional Generative Latent Optimization (cGLO). Our model learns to synthesize completely new samples for every class just by interpolating between samples in the latent space. The proposed method samples the learned latent space using spherical interpolations (*slerp*) and generates a new sample using the trained generator. Our empirical results show that the new sampled set is diverse enough, leading to improvement in image classification in comparison to the state of the art, when trained on small samples of CIFAR-100, CUB-200 and a novel agricultural data *'Hazera'*.

Keywords. Small data, generative models, latent optimization, classification, augmentation

Dedication and acknowledgements

There are a number of people without whom this thesis might not have been written, and to whom I am wish to dedicate this work. To my advisor, Professor Daphna Weinshall, who has given me the opportunity to pursue my ideas through this research. I enjoyed our weekly meetings and our brainstorms. I learnt a lot from this process and I am grateful for that.

To my family, who have endlessly encouraged and supported me throughout this journey.

And most of all, to my lovely wife, Inbar, who was by my side during this long journey and always supported me with great encouragement. Importantly, she also made sure I never forgot to take a break from writing when I needed to have some fun.

Table of Contents

List of Tables

Table **Page**

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Modern deep Convolutional Neural Networks (CNNs) define the state of the art in image classification, as well as many other problems in a wide range of applications. Typically enormous amounts of labeled data are used to train the networks. It is not obvious whether this success can be replicated in domains where the resource of labeled data is not widely available. While hardly unexplored, the question of learning from a small sample remains a very important and challenging problem, not least so in the context of deep learning and image classification. The differences (sometimes subtle) between the general problem of learning from a small sample, and the related problems of few-shot learning and learning from imbalanced datasets, are briefly discussed in the next section.

Different approaches have been explored to address this problem, as reviewed in the next section. The problem can be alleviated by imposing a strong prior on the model, which is less relevant in the context of deep learning, and by employing regularization. Data augmentation may help, reflecting the availability of some prior knowledge about the data. Methods employing semi-supervised [44] and transductive learning [56] make use of unlabeled data, when available. Self-training approaches can also boost performance when labels are scarce. Finally, one may compute a generative model from the training data, and use it to generate new samples. This is the approach we explore in this study. The different approaches are not mutually exclusive and can be used jointly to further boost performance.

Using generative models to augment the training sample is very appealing, especially

at present when very powerful deep generative models are becoming available. The problem is that in general, these models require a very large (possibly unlabeled) sample to achieve effective training, and therefore can only be used to augment the training set in the semi-supervised scenario. To avoid this difficulty, our method leverages a recent generative model - GLO [10], which can be trained effectively from a small sample. This generative method is as effective as adversarial models for image reconstruction, though it is not as effective for the generation of realistic-looking new images.

More specifically, the architecture we propose is based on the GLO model described in Section 3.1.2. GLO learns a latent space code for each data point separately. To improve the properties of the latent space, pushing examples from the same class to lie close to each other and as much as possible separated from the remaining points, we add a classifier to the basic architecture, which is trained using the known multi-class labels of the data. We note that training is not adversarial, and therefore this classifier is not equivalent to the discriminator in the GAN (Generative Adversarial Network) architecture. Additional modifications include the concatenation of noise to the latent space vector. The full model is described in Section 3.1.

The modified architecture allows us to effectively sample specific areas in the latent space and synthesize novel images as explained in Section 3.1.2. In this way, our method bears some similarity to [12], a method designed to deal with the problem of imbalanced datasets by creating synthetic minority class examples. The empirical evaluation described in Section 4.2 shows the success of our model in improving classification performance while training with a small sample, especially in the extreme conditions where the sample size is very small indeed.

The rest of the study is organized as follows. In Section 3.1 we describe cGLO, a new method for data synthesis which can be effectively trained from a small number of labeled points from each class. Our method can make use of unlabeled data when available, further benefiting from additional data in the semi-supervised and transductive learning scenarios. In Section 4.2 we describe how synthetic images are used to boost the training of a discriminative deep classifier. In Section 4.4 we demonstrate the superior performance of our method when compared to alternative methods under extremely low sample conditions. We provide an ablation study and further investigate alternative design choices and their effect on classification performance. The empirical evaluation of our method is described in Section 4.2, using a few datasets and different network architectures that have been used in previous work when using these specific datasets. Our method achieves significant improvement in the task of small data classification.

1.1 Contributions and Outline

Our Contribution in this study is as follows:
I) In Chapter 3 we propose a novel data agnostic method for data generation in the low data regime.
II) In Chapter 4 we show an empirical comparison of our model and other related methods on three datasets. Including two experiments with a real-world novel dataset of agricultural data.
III) Next, we analyze the latent space of the proposed model cGLO and we discuss how different design choices of our model affect the classification performance. Then, we examine the effectiveness of adding a classifier to the latent optimization training process as a regularization method.
IV) Section 4.4.1 describes how the latent space of a generative model is influenced by transductive learning.
V) Finally, section 4.6, investigates the borders of small data learning and semi-supervised learning. We provide a quantitative evaluation using various challenging setups.

CHAPTER 2

REALTED WORK & BACKGROUND

In the first part of this chapter, we survey the different approaches and methods for learning from small data. Next, we give a solid background of the topics discussed through our work.

2.1 Related Work

Two decades ago, the problem of learning from a small sample received considerable attention mainly in the context of Bayesian inference, see for example [6, 18, 55]. Bayesian methods rely on the existence of a large dataset which is correlated with the test set. Under this assumption, a prior can be established and used to identify test examples from unseen classes. These methods have been revisited in recent years and largely investigated in the context of *few-shot learning* or *k-shot learning*. In a commonly used setup of *few-shot learning*, at train time there are many classes with many labeled examples from each class. At test time, some novel classes (usually 5) with only a few examples per class are given, alongside query images from the same novel classes, we elaborate on this topic in section 2.2.2. In contrast, our method is not based on transfer learning, but when additional transfer data is available, it may benefit from working in conjunction with a method that does.

2.1.1 Data Augmentation

A very useful and common technique to overcome the small sample problem is *data augmentation*, as discussed by [54]. This include augmentation in the image domain for image classification [e.g., 3, 14, 34], time-series transformations for natural language processing tasks [e.g., 32, 64], speech recognition [e.g. 45], and machine translation [e.g., 17, 59].

For many learning tasks, acquiring a sufficient amount of training data for training a deep neural network can be difficult. As an example, in medical imaging, the acquisition of labeled training data is very time-consuming and costly, since a trained expert needs to manually annotate every image in the training set. An insufficient amount of training data can lead to overfitting, as the neural network is more likely to just memorize certain aspects of the training set.

Data augmentation is a very common technique to mitigate the overfitting issue. Data augmentation describes the process of generating additional training data by transforming the given input training data. Usually, data augmentation in computer vision tasks is done online, therefore the input image is transformed directly before being fed into the network.

Data augmentation is an effective technique to increase both the amount and diversity of data by randomly augmenting it in the image domain [4, 35, 52]. Common augmentations include translating the image by a few pixels, adding Gaussian noise, flipping the image horizontally/vertically and more.

Intuitively, data augmentation is used to teach a model about invariances in the data domain: classifying an object is often insensitive to horizontal flips or translation. Even for large datasets such as ImageNet [36], it has been shown that data augmentation can be beneficial as an additional regularizer for very deep architectures [48]. Additionally, data augmentation allows for an easy way to incorporate prior knowledge about possible unseen data. For example, if test images are taken with a varying amount of brightness, it might make sense to augment with random intensity shifts to accommodate for the variation in brightness in test data. Possible data augmentation schemes range from simple additive or multiplicative image modifications such as intensity shifts to geometric transformations such as rotation, scaling, elastic deformation to synthetic data generation.

In the following literature review, we divide the prior work into two complementary directions: augmentation based on assumed invariance in the data, and the generation of

a new sample using generative models. Metric learning can also be used for this purpose [e.g., 24, 39], where recently [5] showed that using the cosine distance may improve learning from a small sample.

2.1.2 Invariance-Based Augmentation

As we describe above, common augmentation techniques for image classifications include translating the image by a few pixels, adding Gaussian noise, and flipping the image horizontally or vertically. These techniques require prior knowledge of the data. For instance, vertical flip augmentation of the digit 6 will change its semantic meaning to 9. Methods such as AutoAugment [14] as well as Fast Augment [22, 40] and Smart Augmentations [38], achieve further improvement by searching for the optimal set of augmentations in a predefined set of classical transformations, including random crop, flip, rotation, scale, and translation.

AutoAugment selects the best policies using reinforcement learning directly on the given dataset. The authors defined policy as a combination of several augmentations and they apply the best policy on the dataset. The major drawback of this method lays in the enormous search space. There are around six-teen classic augmentations that have infinite permutations considering their parameters. Therefore, it is needed to limit the search space of the policies into only two operations with a specific set of parameters; despite these limitations, it takes weeks to search for the optimal policy. In contrary, the training time for our proposed model is very short (two hours for 60k samples in low resolution) and the augmentations are not limited in the search space to a predefined set of basic image manipulations The major drawback of such methods lies in the enormous space that needs to be searched, and the huge resources that are needed to perform an effective search.

Additional augmentation methods in image classification include *Cutout* [16], which randomly masks a square region in an image at every training step and thus affects the nature of the learned features, and *Random Erasing* [65], where similarly to dropout randomly chosen rectangular regions in the image have their pixels erased or replaced by random values. *MixUp* [62] uses Alpha-blending of two images to form a new image while regularizing the CNN to favor a simple linear behavior in between training images. *MixMatch* [9] augments MixUp by self training, generating "guessed labels" for each unlabeled example. We note that these methods are not always effective on very small datasets, and may even degrade classification performance as shown in Table 4.1.

2.1.3 Data Augmentation with Generative Models

As alluded to above, generative models can be a powerful tool for data augmentation by making it possible under ideal conditions to sample new examples. Most of the methods which rely on generative models target the few-shot learning scenario. Among these methods, Delta-Encoder [50] investigates the use of a modified AutoEncoder: the model gets two examples from a known class and then learns the 'delta' between the two examples to generate new examples for novel classes. DAGAN [1] employs a GAN composed of a U-net generator and a DenseNet discriminator. [63] modified the discriminator of a semi-supervised GAN to return $2N$ outputs, where the first N elements denote class probability, and the remaining N outputs denote the probability that the example is fake. Methods such as [15, 41] also make use of Auto-Encoders. When using an Adversarial AutoEncoder (AAE), one can populate "dead zones" in the latent space by initializing it uniformly and then applying MixUp [62]. This method requires a large amount of (labeled or unlabeled) data to be effective.

All of these methods try to leverage generative models by sampling new synthetic examples from the learned distribution of the data. However, the estimation of the latent space for a given dataset is challenging and requires many training examples. In the low-shot case, the estimated data distribution is less reliable, and it is, therefore, necessary to rely on the joint distribution of related classes. Thus one approach lets the learning model share its parameters across all classes, using the same generator in a GAN architecture, or the same encoder and decoder in an Auto-Encoder. In the small sample case, the few labeled examples do not represent the true data distribution very reliably, resulting in poor generalization and low-quality synthetic data.

All in all, although GAN models have achieved tremendous success in generating realistic novel images [11, 30], GAN based models have two major drawbacks that make them hard to use for data augmentation. The first is their sensitivity to hyperparameters when trying to reach the Nash equilibrium in the training process, and avoiding mode collapse. The second is the dependency on very large datasets [21].

2.2 Background

2.2.1 Generative Models

In general, a generative model is a powerful way of learning any kind of data distribution both in a supervised and unsupervised manner and it has achieved tremendous success

in just a few years. All types of generative models aim at learning the true data distribution of the training set to generate new data points with some variations. Theoretically, in this work, we use the assumption that for any given images datasets there is a low dimensional structure where the high dimensional distribution concentrates on or near a low dimensional manifold,[7, 47]. It implies that a set of natural images does not span a linear subspace in the pixel space, instead, it is assumed to constitute a low dimensional sub-manifold [8].

In this study, we explored new ideas of variations of GLO 3.1.2, however, it is essential to understand other popular generative models (GAN, VAE) to emphasize the reason choosing the base model in our algorithm 3.1.

Generative Adversarial Networks (GAN)

In 2014 Goodfellow et al.[20] proposed a generative model, called Generative Adversarial Networks (GANs). The classic GANs composed of two networks, a generator denote by $\mathscr{G}$ and a discriminator denotes by $\mathscr{D}$. The discriminator tries to distinguish fake images from real ones while the generator produces fake images when it tries to fool the discriminator. Both networks are jointly trained competitively. The resulting generator can synthesize plausible images. The objective function of a two-player minimax game can be described by the value of the function $V(G,D)$as written in Eq 1:

$$\min_G \max_D V(D,G) = \mathop{\mathbb{E}}_{x\sim\mathscr{P}_{Data}}[\log D(x|y)] + \mathop{\mathbb{E}}_{z\sim\mathscr{P}_{Z(z)}}[\log(1-D(G(z|y)))] \tag{2.1}$$

In other words the generator $\mathscr{G}$ maps a noise vector z in the latent space to a generated image $\mathscr{G}:\mathscr{G}(z))\mapsto\mathbb{R}^{|x|}$ where $z\in\mathbb{R}^{|z|}$ is a sample from the latent space and $|\cdot|$ denotes the image dimension. The discriminator defined by $\mathscr{D}:\mathscr{D}\mapsto(0,1)$ when $x\to 0$ means the discriminator classifies the input image as fake ($x\sim\mathscr{P}_{Z(z)}$) or $x\to 1$ if the input image classified as real ($x\sim\mathscr{P}_{Data}$) [13].

Conditional Generative Adversarial Networks (cGAN) was introduced by Mirza and Osindero[42]. The cGAN is an extension of GAN into a conditional model where the generator $\mathscr{G}$ and the discriminator $\mathscr{D}$ are now conditioned by extra information y. The extra information could be class labels, text, or sketches. cGANs provide additional controls on which kind of data are being generated, while the original GANs do not have such controls. We can perform the conditioning by feeding y into both the discriminator

and generator as the additional input layer. Note, that our model cGLO does not use the same explicit conditioning as cGAN uses. This kind of conditioning derogates the results on small data.

Variational Auto-Encoder

The idea of Variational Autoencoder (VAE) [31], is deeply rooted in the methods of the variational bayesian and graphical model and less similar to a classic AutoEncoder.

Instead of mapping the input into a fixed vector, the objective is to map it into distribution. Denote a distribution p_θ, parameterized by Θ. The relationship between the data input x and the latent encoding vector z can be fully defined by: Prior $P_\theta(z)$, Likelihood $p_\theta(x|z)$ and Posterior $P_\theta(z|x)$. In a nutshell, VAE minimizes minus the sum of the expected log-likelihood (the reconstruction error) and a prior regularization term:

$$\mathscr{L}_{\text{VAE}} = -\mathbb{E}_{q(x|z)} = \left[\log \frac{p(x|z)p(z)}{p(z|x)}\right] = \mathscr{L}_{\text{pixel likelihood}} + \mathscr{L}_{\text{prior}} \tag{2.2}$$

where

$$\mathscr{L}_{\text{pixel likelihood}} = -\mathbb{E}_{q(x|z)}[\log p(x|z)] \tag{2.3}$$

and

$$\mathscr{L}_{\text{prior}} = D_{KL}(p_{z|x}||p(z) \tag{2.4}$$

D_{KL} is the Kullback-Leibler divergence.

2.2.2 Few-Shot Learning

In the previous chapters, we mention several works of the few-shot learning methodology. The following section elaborates on some of the most popular works of this approach. We stress the importance of a large and diverse training set in this approach. The most common basic concept here is *transfer learning*, which encouraged us to abandon this research direction. Though, we did not use directly the few-shot technique we inspired by these ideas in our work.

Problem Statement

The few shot paradigm was first introduce by Vinyals [57], in high level Few shot learning is a synthetic task that aims to distinguish between K labeled samples from

each of the N classes when N is very small (also known as K-shot N-way classification). Formally, given the distribution of tasks $P(\tau)$, a sample task τ from $P(\tau)$ is given by the joint distribution $P^{\tau}_{X\times Y}(x,y)$, where task is to predict y given x. In total, there is a set of tasks $\{\tau\}_{i=1}^{N}$, each training task τ is a tuple $\tau = (S_\tau, Q_\tau)$ where the support set is denoted as $S_\tau = S^s_\tau \bigcup S^u_\tau$, and the query set is denoted as $Q_\tau = Q^s_\tau \bigcup Q^u_\tau$. The supervised set $S^s_\tau = \{(x_1,y_1),\ldots(x_{N\times K},y_{N\times K})\}$ and the query set is $S^u_\tau = \{x_1,\ldots x_M\}$ contains unlabeled samples from the N classes; Q^s_τ and Q^u_τ are the query sets that are defined similar to S_τ. The objective of the model is to predict the query set labels given the support set.

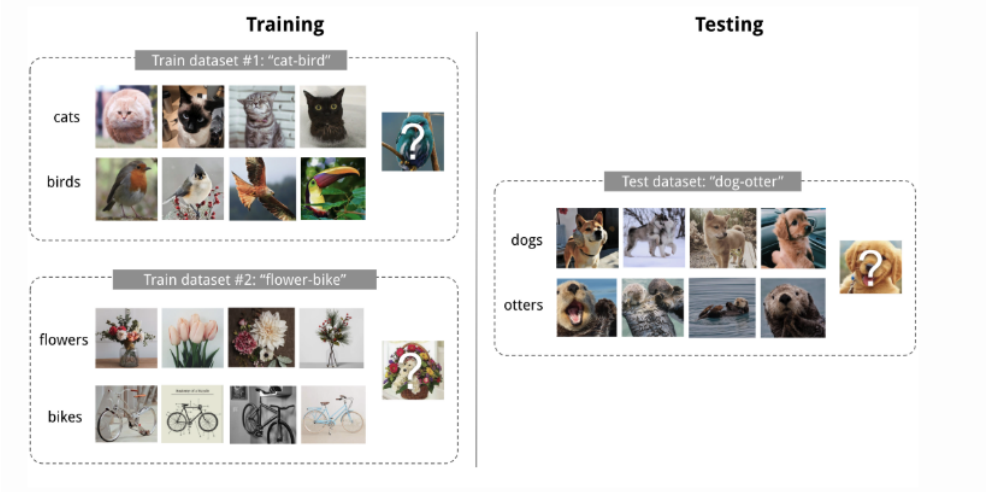

Figure 2.1: Few Shot learning. An example of a task τ of 4-shots 2-classes (image source: Pinterest)

Metric Learning

The very first works in the few-shot learning approach are *Matching Networks* [57] and *Prototypical Networks* [26, 53]. Both of these methods based on the metric in the latent space between the give samples.

Matching Networks. The basic idea is to use a good representative latent space and to compare the support set S_τ to the query set Q_τ in the latent space using attention kernel,

$$a(x,x_i) = \frac{\exp\cos f(x), f(x_i)}{\sum_{j=1}^{k} \exp\cos g(x), f(x_i)} \tag{2.5}$$

Where the attention kernel depends on two embedding functions, f and g, for encoding the query sample and the support set samples respectively. The attention weight

between two data points is the cosine similarity $\cos(\cdot)$, between their embedding vectors, normalized by softmax.

Prototypical Networks. Use an embedding function f to encode each input into a d-dimensional feature vector. A prototype feature vector is defined for every class $c \in C$, as the mean vector of the embedded support data samples in this class,

$$M_c = \frac{1}{|S_c|} \sum_{x_i \in S_c} f(x_i) \tag{2.6}$$

Now the loss function is defined as negative log-likelihood: $\mathscr{L}_\theta = -\log(\mathbb{P}(y = c|x))$ when

$$\log(\mathbb{P}(y = c|x)) = \text{softmax}(-d(f(x), M_c) = \frac{\exp(-d(f(x), M_c))}{\sum_{c' \in C} \exp(-d(f(x), M'_c))} \tag{2.7}$$

d is the distance function (in the paper they used the L_2 metric).

Optimization-Based

While deep learning models learn through the backpropagation of gradients, the gradient-based optimization is neither designed to cope with a small number of training samples nor to converge within a small number of optimization steps. The basic idea of this family of models is to adjust the optimization algorithm so that the model can be good at learning with a few examples [e.g., 19, 43]. The idea is to use two optimizers, when the outer optimizer is updated by the weighted average of the inner optimizer which learns on many small tasks $\tau \sim S_\tau$.

Model-Based

Meta-learning systems are trained by being exposed to a large number of tasks and are then tested in their ability to learn new tasks. An example of a task might be classifying a new image within 5 possible classes, given one example of each class, or learning to efficiently navigate a new maze with only one traversal through the maze. This differs from many standard machine learning techniques, which involve training on a single task and testing on held-out examples from that task.
In [49] the authors propose a method based on memory. There is a controller (LSTM) that remembers the previous training example to learn patterns in one-shot learning. Also, they talked about combining curriculum learning - first to train one class and every few thousands of iterations increase by one the number of classes that the model sample the train set.

CHAPTER 3

OUR METHOD

In this chapter, we present our novel modification to Generative Latent Optimization (GLO) which we use to synthesize new images to augment a given data set. Moreover, we compare the performances of a GLO-based method to a standard data augmentation in classification accuracy.

3.1 Conditional GLO for Small Data Augmentation

We propose a method for multi-class image classification from small sample, which consists of data augmentation with a generative model. Specifically, in order to use modern deep learning classifiers, which typically require large amounts of training data, we augment the small training set by sampling from a generative model. Our generative model, called cGLO, is based on the GLO architecture [10] which can be trained effectively from a small sample. This architecture is modified so that it can benefit from both labeled and unlabeled data. With access to only small amounts of unlabeled data or none at all, our results surpass the state of the art. The code is available online[1].

3.1.1 Model Overview

The generative model we use to sample new data includes the basic GLO architecture with generator G_θ, and an added small classifier f_ϕ which is trained to classify the labeled

[1] https://github.com/IdanAzuri/Learning-from-Small-Data

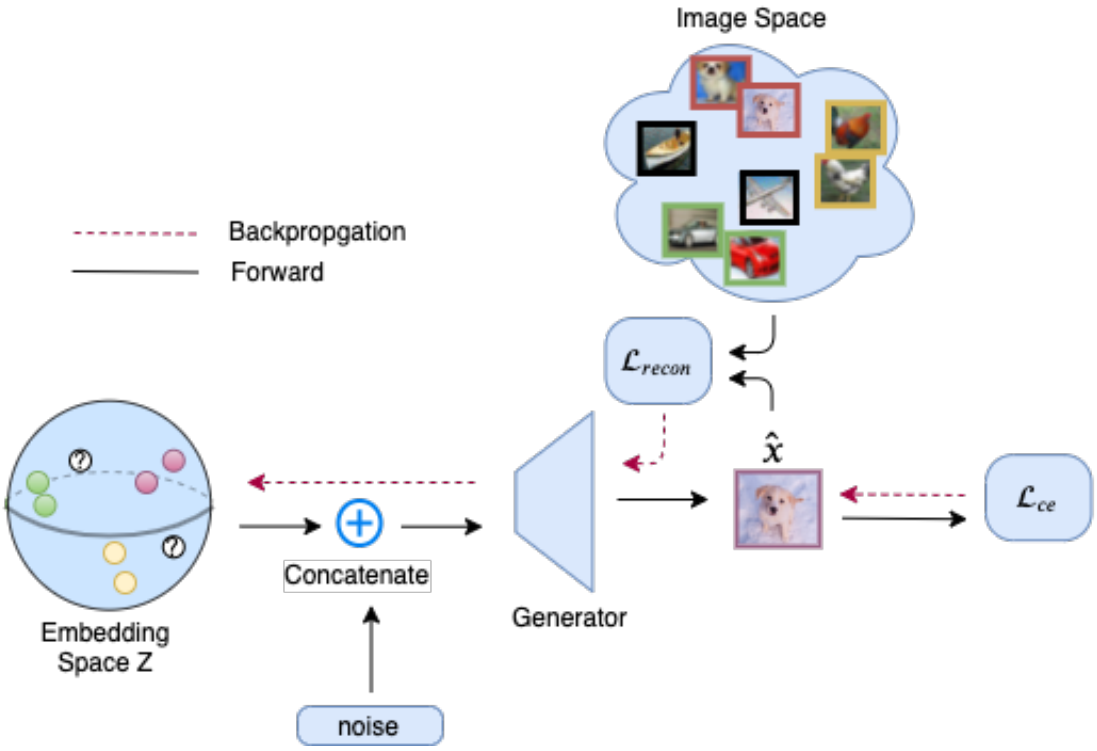

Figure 3.1: Schematic illustration of our method. Every $\mathbf{z}_i \in Z$, where Z denotes the unit sphere, is mapped to a specific image x_i in the image space. In this illustration, the color of each image frame marks the class label of the image (pink:dog, green:cars, yellow:bird), while black frames mark unlabeled images. The classifier is used to propagate error only when a label is given.

data (see Fig. 3.1). Training is not adversarial, and therefore this classifier is nothing like the discriminator in the GAN (Generative Adversarial Network) architecture. As in [10], the latent space is initialized randomly $\{\mathbf{z}_i \in Z\}$ where Z is the unit sphere in $\mathbb{R}^d$. Every vector $\mathbf{z}_i$ is mapped to an image $\{x_i \in X | x_i \in \mathbb{R}^{3\times H\times W}\}$ from the given (small) training set.

The training process has two modes: With unlabeled data, the reconstruction loss in (3.2) is used to train G_θ just like the original GLO model. With labeled data, the reconstruction loss is augmented with the cross-entropy loss corresponding to the loss of the added classifier f_ϕ (see Fig. 3.1).

Here we use the perceptual loss [28] to measure the reconstruction loss. More specifically, in order to compute the perceptual loss we extract the activation vectors in layers

Algorithm 1 cGLO. The algorithm optimizes the reconstruction loss $\mathscr{L}_{percep}$ of generator G_θ, and the cross entropy loss $\mathscr{L}_{ce}$ of discriminator f_ϕ.

Input: unlabeled data P_{D_u}, labeled data P_{D_L}, γ, epochs
epoch = 0
Initialize $\{\mathbf{z}_i\}_{i=1}^n$ where $\{\mathbf{z}_i \in Z : ||\mathbf{z}_i||_2 = 1\}$
repeat
  for $(x_i, y_i) \in P_{D_L}$ **do**
    $\mathscr{L}_{percep} = \sum_j \lambda_j ||\xi_j(G_\theta(z_i)) - \xi_j(x_i))||_1$
    $\mathscr{L}_{ce} = \mathscr{L}_{CE}(f_\phi(G_\theta(z_i)), y_i)$
    $\mathscr{L} = \mathscr{L}_{percep} + \gamma \mathscr{L}_{ce}$
    Update $\{\mathbf{z}_i\}, \theta, \phi$ using the gradient of $\mathscr{L}$
  end for
  (this part is optional for transductive learning mode)
  for $x_i \in P_{D_u}$ **do**
    $\mathscr{L} = \sum_j \lambda_j ||\xi_j(G_\theta(z_i)) - \xi_j(x_i))||_1$
    Update $\{\mathbf{z}_i\}, \theta$ using the gradient of $\mathscr{L}$
  end for
  $epoch += 1$
until $epoch > epochs$

$conv_{1\_2}$, $conv_{2\_2}$, $conv_{3\_2}$, $conv_{4\_2}$ and $conv_{5\_2}$ of a VGG-16 network. Denoting the output tensor of layer $conv_{j\_2}$ for input image x by $\xi_j(x)$, we compute the difference between the original image and its reconstructed version by:

$$\mathscr{L}_{percep}(x_i, \mathbf{z}_i; \theta) = \sum_j \lambda_j ||\xi_j(G_\theta(\mathbf{z}_i)) - \xi_j(x_i))||_1 \tag{3.1}$$

Above θ denotes the parameters of the generator G, and λ_j the weight of layer j (usually the weighted average).

3.1.2 Generative Latent Optimization

Our method is described in Alg. 1. Its components are described next.

Generative model. Generative Latent Optimization (GLO) [10] is a relatively simple method, relying on a relatively small number of parameters. GLO maps every image x_i from the dataset to a low-dimensional random vector $\mathbf{z}_i$ in the latent space Z. It then passes the random vector into a generator $G_\theta(\cdot)$, which is optimized to minimize the reconstruction loss between $G_\theta(\mathbf{z}_i)$ and x_i.

Formally, let $\{x_1, x_2 \ldots x_n\} \in X$ denote a set of images where $x_i \in \mathbb{R}^{3 \times W \times H}$. Choose n d-dimensional random vectors on the unit sphere $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\} \in Z$ where $Z \subseteq \mathbb{R}^d$. Pair ev-

ery image $x_i \in X$ with a random vector $\mathbf{z}_i \in Z$, to achieve the mapping $\{(x_1, \mathbf{z}_1), \ldots, (x_n, \mathbf{z}_n)\}$. Finally, learn jointly the parameters θ of the generator $G_\theta : Z \rightarrow X$, where the optimal set $\{\mathbf{z}_i\}$ and parameters θ are obtained by minimizing the following objective:

$$\min_{\theta} \sum_{i=1}^{n} \left[\min_{\mathbf{z}_i \in Z} \mathscr{L}_{percep}(x_i, \mathbf{z}_i; \theta) \right] \quad s.t. \quad ||\mathbf{z}_i||_2 = 1 \tag{3.2}$$

The loss $\mathscr{L}_{percep}(x_i, \mathbf{z}_i; \theta)$, defined in (3.1), measures the reconstruction loss between $G_\theta(\mathbf{z}_i)$ and x_i. We note that while in the original GLO the Laplacian pyramid loss is used instead of $\mathscr{L}_{percep}$, the minimization of $\mathscr{L}_{percep}$ appears to yield more realistic results [25].

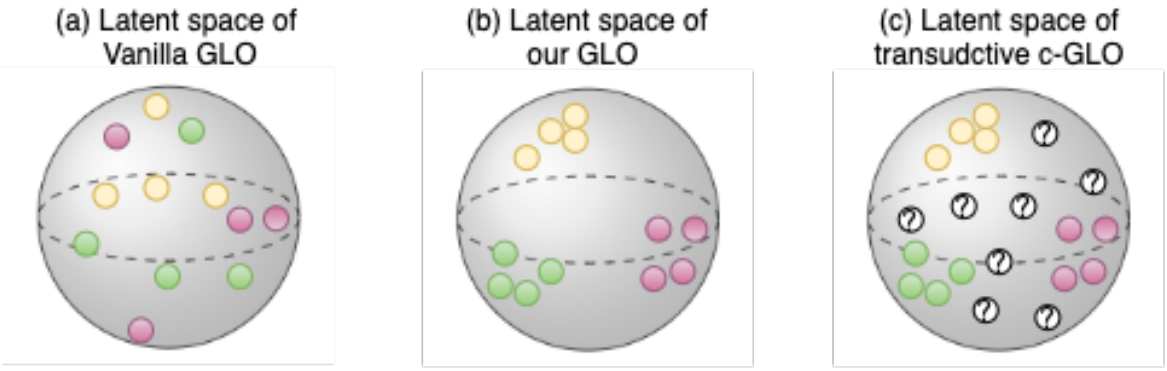

Figure 3.2: Illustration of the latent space Z in our method vs. vanilla GLO. (a) Vanilla GLO: vectors $\mathbf{z}_i \in Z$ do not have a semantic meaning in Z. (b) Our method: vectors from the same class are grouped. (c) Our method in transductive mode. Notations: filled colored circles represent different labeled data points, where color corresponds to class identity. Black circles with the symbol "?" represent unlabeled data points.

Adding a Classifier. Possibly the main weakness of using GLO as a generative model is the relatively low quality of images generated when sampling new points in the latent space Z. The problem lies in the sparsity of the learned set $\{\mathbf{z}_i\}_{i=1}^n$, which lacks structure since each $\mathbf{z}_i$ is trained independently. In order to decrease the intra-class distances and increase the inter-class distances in the latent space representation, we propose the conditional model cGLO. In this model, the generator G_θ is augmented by a weak classifier f_ϕ, which is trained to classify the labeled data. When the label of x_i is known, $\mathscr{L}_{percep}$ in (3.2) is replaced by $\mathscr{L}_{percep} + \mathscr{L}_{ce}$, where $\mathscr{L}_{ce}$ is the cross-entropy loss of classifier f_ϕ.

Sampling the Latent Space. Generating images based on randomly sampling the latent space, even when restricted to the immediate vicinity of $\{\mathbf{z}_i\}_{i=1}^n$, still produces low quality somewhat meaningless images. Therefore, instead of randomly sampling Z, we generate new image codes by interpolating between the known latent vectors $\{\mathbf{z}_i\}_{i=1}^n$. Since the latent space Z is a hyper-sphere, we employ to this end *spherical linear interpolation* (*slerp*) [51], which is defined as follows:

$$slerp(q1, q2; t) = q_1 \frac{\sin(1-t)\vartheta}{\sin\vartheta} + q_2 \frac{\sin t\vartheta}{\sin\vartheta} \tag{3.3}$$

Above $t \in [0,1]$, and ϑ is the angle between q_1 and q_2, computed as $\vartheta = \cos^{-1}(q_1 \cdot q_2)$.

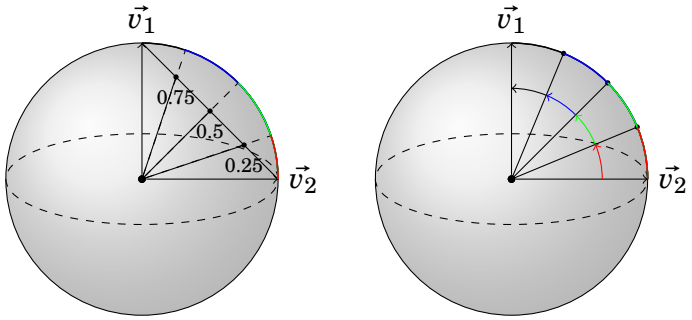

Figure 3.3: *Slerp* vs. *lerp*. Left side: linear interpolation between $\vec{v_1}$ and $\vec{v_2}$ with $t \in [0.25, 0.5, 0.75]$. Right side: spherical interpolation. Note that both the length and arc length of the interpolated vectors are equal in *slerp* but unequal in *lerp*.

As shown in Fig. 3.3, in *slerp* interpolation follows the great circle path on an d-dimensional hyper-sphere (with elevation changes) between two points $\mathbf{z}_i$ and $\mathbf{z}_j$. This technique has shown promising results in the context of both VAE and GAN generative models and with both uniform and Gaussian priors [60].

Why *slerp*? Linear interpolation (*lerp*) is the simplest method to traverse the latent space manifold between two known locations. Often it is used to show inartistic learned features that capture the semantics of the dataset [e.g., 29, 35]. However, [2] noted that linear interpolation in the latent space is often inappropriate since the latent spaces of most generative models are embedded in high dimensional spaces (over 50 dimensions). In such a space, linear interpolation traverses locations that are extremely unlikely given the prior, whether Gaussian or uniform.

Noise concatenation. Training from a small sample is more susceptible than ever to random perturbations in the data. To increase training robustness, we concatenate noise

to the latent vector such that the input to the generator G_θ is $[\mathbf{z}_i, \varepsilon] \quad \varepsilon \sim N(0, \sigma I)$, see Fig. 3.1. In effect, this introduces randomness into the training via the batch normalization layers [27].

CHAPTER 4

EXPERIMENTAL EVALUATION

In this chapter, we empirically evaluate our method cGLO and other augmentation methods from different approaches as discussed in section 2. We showed our investigation of the proposed method variants on different setups. We demonstrated the superiority of the model on the small data regime (10-100 examples per class) while keeping the training time very short compared to other methods.

4.1 Datasets

We evaluate our method on three datasets composed of two standard benchmarks for image classification and one real-world dataset of cucumbers collected by *Hazera*. The first dataset is CIFAR-100 [33], which includes $50,000$ 32×32 color images, with 100 classes and 500 images per class. The relatively small size of the images allows us to perform an exhaustive ablation study on this dataset as described in Section 4.4. The second dataset is CUB-200 [58], which includes high-resolution fine-grained images of 200 species of birds, with only 30 images per class. This makes this dataset a more appropriate testbed for a method that addresses the small sample problem.

***Hazera* (Cucumbers)**

An extremely small dataset of greenhouse cucumbers in different illness severity levels from one (sickest) to nine (healthy). In this study, we divided the data into classes by the plant's health condition, where labels one to three are grouped to the sick group

and labels seven to nine are grouped to the healthy group. The training set contains 100 examples from both of the groups and the test set contains 28 examples equally distributed. This small data is very unique and challenging to classify as it has not been collected for classification purposes. It can be seen in Figure 4.2 the difficulty in distinguishing between the classes even in human eyes.

4.2 Experimental Protocol

For each benchmark, we defined a small-sample task by sub-sampling the original training set of the corresponding dataset. To allow for comparison with other methods, the subset splits were adapted from [5]. As classification engine we used, unless otherwise noted, the *WideResNet-28* model [61] for CIFAR-100, and the Resnet50 model [23] for CUB-200. Baseline results were obtained by training the corresponding model using the training set with only standard data augmentation.

When using our method, we augmented the training set using cGLO. Specifically, we start by sampling a mini-batch from the training data in each SGD optimization step. Each example x_i in the mini-batch is used for training with a probability of 0.5. Otherwise (with probability 0.5) it is replaced by a new image obtained by sampling the latent space $G(slerp(\mathbf{z}_i, \mathbf{z}_j, t))$. $\mathbf{z}_j$ is the latent representation of some example from the same class c as x_i, sampled uniformly from the latent codes of all remaining examples in class c. The *slerp* interpolation factor is sampled uniformly from the set $[0.1, 0.2, 0.3, 0.4]$.

We compared our results with state-of-the-art methods that are suitable for the small sample domain, using as much as possible public-domain code. Thus we compared sample augmentation with cGLO to image augmentation with Cutout [16], Random Erase [65] and MixMatch [9]. In each case we repeated the same procedure as described above, replacing the generation of a new image using cGLO by an image obtained from the corresponding augmented set of images. We also evaluated the method described in [5], which was explicitly designed to handle small sample, using code provided by the authors.

4.3 Implementation Details

We used SGD optimization with learning rate 0.1 for CIFAR-100 and 0.001 for CUB-200, with a batch size of 128 and 16 respectively. In all the experiments we used the standard categorical cross-entropy loss function when training the weak classifier $f\phi$. (We note

in passing that using a strong classifier decreased the quality of the generated images and harmed the final performance.) Images from the CUB-200 dataset were resized to 256 pixels wide in their smaller side, and then randomly cropped to 224×224 pixels. As stated above, in all the experiments (including baseline) we adopted the standard transformations of random horizontal flipping and random crop for data augmentation. The settings for *Hazera* are the same as used for CUB-200.

In all cases, the latent space Z was the unit sphere in $\mathbb{R}^{128}$. For the generator, we used a standard off-the-shelf DCGAN architecture [46]. More modern GAN architecture can be readily used instead and improve the reconstruction quality. However, it is worth noting that our method can improve the SOTA while using a relatively simple GAN architecture. We trained the model on 2 × Tesla P100 GPUs for 200 epochs; every epoch took around 40 seconds on CIFAR-100 and 3 minutes on CUB-200.

To achieve uniformity we over-sampled the training set in proportion to the size of the training set. For example, with 50 samples per class in CIFAR-100, we trained the model $\times 10$ iterations. Standard errors (STE) were obtained from 3 runs with different seeds in all study cases except for MixMatch, where a single result is reported since each MixMatch run took a very long time.

4.4 Results

CIFAR-100 and CUB-200 The results of our empirical study are summarized in Table 4.1 for CIFAR-100, Table 4.2 for CUB-200. We used small sample partitions, with 10 to 100 labeled images per class in CIFAR-100, and 5 to 30 labeled images per class in CUB-200. We compared our model to three different methods of data augmentation and one small sample method.

SPC	BASELINE	CGLO	MIXMATCH	CUTOUT	RANDOM ERASE	[5]
10	22.89 ± 0.09	**28.55±0.40**	24.8	23.43±0.24	23.26±0.27	23.01 (22)
25	38.39 ± 0.18	**43.84±0.25**	40.17	39.11±0.59	37.45±0.15	28.05 (35)
50	47.82±0.11	**52.95±0.20**	49.87	52.11±0.28	50.50±0.41	44.55 (48)
100	61.37±0.13	64.27±0.04	59.03	**64.49±0.10**	64.03±0.22	55.99 (58)

Table 4.1: Comparison of Top-1 Accuracy (including STE) for CIFAR-100 using WideResnet-28, with a different number of training SPC (Samples Per Class). The methods used for comparison are described in the text below, where caveats regarding the results reported in the last column are also discussed. Best results are marked in bold.

Figure 4.1: CIFAR-100 with *10 samples* per class. Each row shows six new images generated based on smooth interpolation in the latent space between two reconstructed images.

Fig. 4.3 illustrates the quality of the reconstruction when using cGLO, showing a relatively precise though somewhat blurred reconstruction. Fig. 4.4 illustrates the kind of synthetic images we get while relying on a very small sample. Note that since the purpose of generating new images is to boost classification from small sample, low image quality does not preclude their usefulness for the task.

The two augmentation methods used in our comparisons, Random Erasing [65] and Cutout [16], achieve a relatively good performance; with CIFAR-100 and a 100 samples per class, Cutout achieves the highest accuracy (similar to cGLO) in our experiments. Nevertheless, in all the other small sample cases we have studied cGLO significantly outperforms these methods.

MixMatch [9] is a new technique that achieves state-of-the-art results on multiple datasets in a semi-supervised setting. In the supervised small sample regime, this method does not perform very well as can be seen in Tables 4.1 and 4.2, but see Section 4.4.4.

SPC	BASELINE	CGLO	MIXMATCH	CUTOUT	RANDOM ERASE	[5]
5	50.79±0.19	**51.52±0.21**	15.01	50.63±0.31	48.90±0.45	17.80 (35)
10	64.11±0.22	**65.13±0.12**	36.02	64.33±0.02	63.72±0.20	34.23 (60)
20	69.11±0.55	**74.16±0.17**	60.57	68.47±0.20	66.14±0.23	52.00 (76)
30	75.15±0.10	**77.75±0.20**	70.41	74.97±0.34	73.74±0.34	62.25 (82.5)

Table 4.2: Comparison of Top-1 Accuracy (including STE) for CUB-200 using ResNet-50, with a different number of training SPC (Samples Per Class). The methods used for comparison are described in the text below, including some caveats. Best results are marked in bold.

Possibly, the blending of images in pixel space, which is intended to provide some means of regularization, is only effective when enough training data is available. Otherwise, it feeds noisy examples to the model and makes it harder to generalize.

[5] describes a distance-based method that is designed to handle the small sample challenge, among other things. The results, when using the code published by the authors in our experimental design, are shown in the last column of Tables 4.1 and 4.2. Admittedly, we were not able to reproduce their published results. We, therefore, show in round brackets our best estimate of their original results, which were only reported in a pictorial format[1].

Real-World Small Data (*Hazera*). We test the effectiveness of our method on raw agricultural data without any pre-processing. The first experiment is on the entire dataset (200 training images) as described in section 4.1. The second experiment is an extremely low regime of 40 training samples. We followed the experimental protocol depict previously for CUB-200 without any further adaption. The results are surprisingly good, even in the experiment of 20 samples per class. Our model, cGLO, outperforms other augmentation methods. While Cutout and Random Erase have not improved generalization of the learned model, cGLO achieved the best results in both experiments. The experiment's outcome is summarized in Table 4.3.

[1]These experiments used Resnet-110 for CIFAR-100 rather than the WideResNet-28 model used here, and Resnet-50 for CUB-200 as in our experiments.

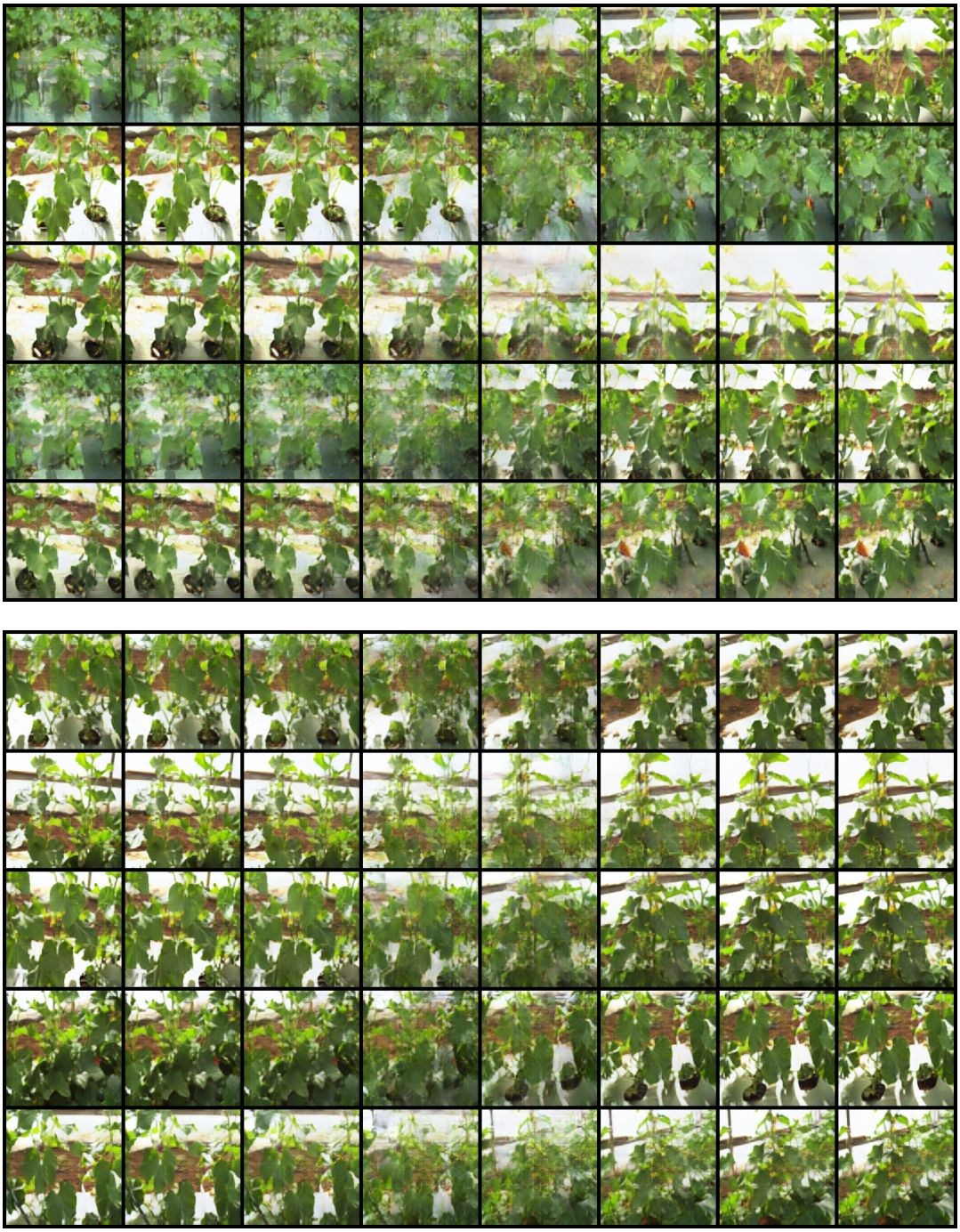

Figure 4.2: *Hazera* dataset, cGLO interpolations. The upper figure shows sick plants, the bottom figure shows healthy plants. Each row shows six new images generated based on smooth interpolation in the latent space between two reconstructed images (the leftmost column and the rightmost column). Note that cGLO trained on 40 examples only.

Model	20 SPC	100 SPC
Baseline	52.65±0.44	53.57±0.47
Cutout	46.10±1.05	51.35±0.33
Random Erase	49.21±0.68	50.00±0.00
cGLO	**53.48±0.56**	**62.65±0.91**

Table 4.3: Comparison of Top-1 Accuracy (including STE) for *Hazera* dataset using ResNet-50. SPC stands for Samples Per Class. Best results are marked in bold.

Figure 4.3: CUB-200 Reconstruction results: upper row shows the original image, second row shows the corresponding reconstructed image. cGLO was trained only on *10 examples* per class.

4.4.1 Ablation Study

In this section, we review and evaluate different design choices used in the architecture and the approach proposed in this work. In this ablation study, we used CIFAR-100 with 25 labeled training examples per class. The results are summarized in Table 4.4.

More specifically, we see in Table 4.4 the effect of omitting different components of cGLO, including classifier f_ϕ, noise concatenation, and replacing *slerp* by vanilla linear interpolation. We note that when omitting classifier f_ϕ, the reconstruction loss achieves a better score, but the augmentation fails to generate *'good'* examples to improve the classification. The *'Baseline'* case shows the results of training without sampling additional images. *'Tranductive'* shows the added benefit obtained from including the

Model	Top 1 Acc.	Top 5 Acc.
cGLO	43.84±0.25	70.73±0.07
Baseline	38.39±0.18	67.77±0.18
No Classifier	41.57±0.54	69.55±0.11
No Noise	43.31±0.02	70.05±0.02
Lerp	43.01±0.06	70.51±0.03
Transductive	44.39±0.12	71.28±0.09

Table 4.4: Ablation Study: Top 1 and Top 5 accuracy is calculated based on the architectural variants described Section 4.4.1.

unlabeled test set in the training of generator G_θ.

4.4.2 Additional Design Choices

In this section, we describe a few alternative design choices that proved less effective, as summarized in Table 4.5.

Latent Classifier. One can optimize the discriminative loss $\mathscr{L}_{CE}$ directly in the latent space using $(\mathbf{z}_i, y_i)$ instead of the image space $(G(\mathbf{z}_i), y_i)$. Here we used a 3 layer fully connected network with inter-layer ReLU activation.

Model	Top 1 Acc.	Top 5 Acc.
Latent Classifier	43.08±0.06	70.22±0.05
Hypercube Init	44.21±0.61	70.89±0.46
ResNet Init	42.95±0.22	70.10±0.13
Additive Noise	41.02±0.43	68.10±0.11
Cosine Loss	41.01±0.27	68.45±0.31

Table 4.5: Top 1 and Top 5 accuracy of additional design choices as explained in the text.

Z **Initialization.** We investigated different ways to initialize the latent space mappings while exploiting some prior knowledge we have on the data, including: **i)** Hypercube vertices: every class is initialized in the vicinity of a different vertex of the hypercube in $\mathbb{R}^{128}$. **ii)** ResNet: each image is assigned the corresponding activation in the penultimate layer of a pre-trained ResNet model.

Additive Noise. cGLO relies on the concatenation of noise to the latent space representation. To investigate the contribution of this mechanism, and following [37], we

Method	supervised only	1000 unlabeled	35k unlabeled
Baseline	38.39 ± 0.18	-	-
MixMatch	40.17	42.39	50.34
cGLO	43.84 ± 0.25	44.52 ± 0.12	44.73 ± 0.07

Table 4.6: Semi-supervised scenario: Top-1 Accuracy of MixMatch vs cGLO when shown 25 labeled examples per class and a varying number of unlabeled examples, where each case corresponds to a different column.

explored a simpler alternative, where random noise $\varepsilon \sim \mathcal{N}(0, \sigma I)$ is sampled i.i.d and added to $\mathbf{z}_i$ before calculating the loss (3.2). The goal is to obtain a better representation of the image manifold by learning the ε ball around every example both in the latent space and the image space. However, as shown in Table 4.5, this approach leads to performance degradation in the final classification.

Cosine Loss. It is argued in [5] that the cosine loss is a better optimization function for the small sample regime. In our experimental setup, the cross-entropy classification loss provided better results, see Table 4.5.

4.4.3 Relation to Classical Augmentations

Next, we investigate the relationship between new images generated by our method and images generated by methods using classical augmentation techniques. To this end we adopt AutoAugment [14], a method designed to find a proxy for the optimal set of classical transformations to augment images in CIFAR-100, searching through 16 types of color-based and geometric base transformations. The case studied here is CIFAR100 with 50 labeled examples per class, and with transductive learning (similar results are obtained without transductive learning).

When using the two methods - AutoAugment and cGLO - in conjunction, each method seems to provide an independent contribution as shown in Table 4.7. From this, we conclude that the contribution of cGLO goes beyond the contribution of augmentation by classical image transformation. Note that AutoAugment is trained on the entire training set of CIFAR-100 so the direct comparison of our method to this policy is not applicable.

AutoAu.	cGLO	Top-1 Accuracy	Top-5 Accuracy
		50.37 ± 0.05	75.61 ± 0.01
	✓	53.35 ± 0.23	77.60 ± 0.12
✓		53.80 ± 0.10	79.18 ± 0.13
✓	✓	56.31 ± 0.02	80.66 ± 0.04

Table 4.7: Top-1 and Top-5 accuracy when augmenting a small dataset by cGLO alone (second row), AutoAugment alone (third row), or both (fourth row). Note that each method boosts performance on its own, while when used in conjunction additional performance boost is seen.

4.4.4 Using Unlabeled Data

While in the fully supervised scenario cGLO outperforms MixMatch as shown in Tables 4.1 and 4.2, MixMatch performs better when given access to unlabeled data, and eventually, it outperforms cGLO as shown in Table 4.6. cGLO can also use unlabeled data to boost the training of the generator G_θ, but as shown in Table 4.6, clearly MixMatch benefits from unlabeled data more considerably. We note that cGLO can benefit from using the test data during training following the transductive learning procedure, which is another way of using unlabeled data to boost training.

CHAPTER 5

SUMMARY AND CONCLUSIONS

5.1 Summary and Discussion

In this work, we revisited the problem of learning from small sample. We developed a deep generative model, conditional GLO (cGLO), which can be effectively trained to generate examples when seeing only a small sample of data. New examples are synthesized by interpolating between the latent vectors of known examples. When using small sample scenarios generated from the CIFAR-100 and CUB-200 benchmarks and also form a novel dataset *'Hazera'*. We show that our method improves classification over the baseline and several alternative methods. Thus our method defines the state of the art in small sample image classification.

Our generative model is based on latent space optimization. Latent optimization does not involve an encoder like some other generative methods (such as the Variational Auto Encoder). In particular, this implies that the number of variables grows linearly with the number of data samples. Contrary to GAN, latent optimization learns every latent representation separately, and therefore it does not require much data in order to achieve decent reconstruction results as demonstrated in Fig. 4.3.

In GLO, the optimization of each representation vector z_i separately also implies that the dimensions of the latent space do not correspond to the semantic features of the data. To address this weakness and to inject some semantic structure into the latent space representation, we added a classifier to the latent optimization training process. Unlike GANs, the classifier is not trained in an adversarial fashion. Rather, we use

the classification loss $\mathscr{L}_{CE}$ over the reconstructed examples $G_\theta(\mathbf{z}_i)$ to induce semantic relations into the latent space and allow for better sampling and new image generation (see Fig. 3.2).

The unique aforementioned properties of our model allow it to improve the training efficacy of deep classifiers in the small sample regime. In this regime, the classifier only sees a small number of labeled examples from each class. We suggest two complementary approaches using our proposed transductive learning option. It can be used in conjunction with our method, and may benefit from unlabeled data in a semi-supervised manner, or unrelated labeled datasets which can be used for transfer learning.

BIBLIOGRAPHY

[1] A. ANTONIOU, A. STORKEY, AND H. EDWARDS, *Augmenting image classifiers using data augmentation generative adversarial networks*, in International Conference on Artificial Neural Networks, Springer, 2018, pp. 594–603.

[2] G. ARVANITIDIS, L. K. HANSEN, AND S. HAUBERG, *Latent space oddity: on the curvature of deep generative models*, 2017.

[3] H. S. BAIRD, *Document image defect models and their uses*, Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR '93), (1993), pp. 62–67.

[4] H. S. BAIRD, K. YAMAMOTO, AND H. BUNKE, eds., *Structured Document Image Analysis*, Springer-Verlag, Berlin, Heidelberg, 1992.

[5] B. BARZ AND J. DENZLER, *Deep learning on small datasets without pre-training using cosine loss*, CoRR, abs/1901.09054 (2019).

[6] J. BAXTER, *A bayesian/information theoretic model of learning to learn via multiple task sampling*, in Machine Learning, 1997, pp. 7–39.

[7] Y. BENGIO, I. J. GOODFELLOW, AND A. COURVILLE, *Deep Learning*, MIT Press, 2015, ch. 17, The Manifold Perspective onRepresentation Learning.
Book in preparation for MIT Press.

[8] Y. BENGIO, G. MESNIL, Y. DAUPHIN, AND S. RIFAI, *Better mixing via deep representations*, in Proceedings of the 30th International Conference on Machine Learning, S. Dasgupta and D. McAllester, eds., vol. 28 of Proceedings of Machine Learning Research, Atlanta, Georgia, USA, 17–19 Jun 2013, PMLR, pp. 552–560.

[9] D. BERTHELOT, N. CARLINI, I. GOODFELLOW, N. PAPERNOT, A. OLIVER, AND C. RAFFEL, *Mixmatch: A holistic approach to semi-supervised learning*, arXiv preprint arXiv:1905.02249, (2019).

[10] P. BOJANOWSKI, A. JOULIN, D. LOPEZ-PAS, AND A. SZLAM, *Optimizing the latent space of generative networks*, in Proceedings of the 35th International Conference on Machine Learning, J. Dy and A. Krause, eds., vol. 80 of Proceedings of Machine Learning Research, Stockholm, Sweden, 10–15 Jul 2018, PMLR, pp. 600–609.

[11] A. BROCK, J. DONAHUE, AND K. SIMONYAN, *Large scale gan training for high fidelity natural image synthesis*, arXiv preprint arXiv:1809.11096, (2018).

[12] N. V. CHAWLA, K. W. BOWYER, L. O. HALL, AND W. P. KEGELMEYER, *Smote: synthetic minority over-sampling technique*, Journal of artificial intelligence research, 16 (2002), pp. 321–357.

[13] A. CRESWELL, T. WHITE, V. DUMOULIN, K. ARULKUMARAN, B. SENGUPTA, AND A. A. BHARATH, *Generative adversarial networks: An overview*, CoRR, abs/1710.07035 (2017).

[14] E. D. CUBUK, B. ZOPH, D. MANÉ, V. VASUDEVAN, AND Q. V. LE, *Autoaugment: Learning augmentation policies from data*, CoRR, abs/1805.09501 (2018).

[15] T. DEVRIES AND G. W. TAYLOR, *Dataset augmentation in feature space*, arXiv preprint arXiv:1702.05538, (2017).

[16] ——, *Improved regularization of convolutional neural networks with cutout*, arXiv preprint arXiv:1708.04552, (2017).

[17] M. FADAEE, A. BISAZZA, AND C. MONZ, *Data augmentation for low-resource neural machine translation*, arXiv preprint arXiv:1705.00440, (2017).

[18] L. FEI-FEI, R. FERGUS, AND P. PERONA, *Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories*, in Learning Generative Visual Models From Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories, 2004.

[19] C. FINN, P. ABBEEL, AND S. LEVINE, *Model-agnostic meta-learning for fast adaptation of deep networks*, in Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 1126–1135.

[20] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative adversarial nets*, in Advances in Neural Information Processing Systems 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds., Curran Associates, Inc., 2014, pp. 2672–2680.

[21] I. J. GOODFELLOW, *Nips 2016 tutorial: Generative adversarial networks*, ArXiv, abs/1701.00160 (2017).

[22] R. HATAYA, J. ZDENEK, K. YOSHIZOE, AND H. NAKAYAMA, *Faster autoaugment: Learning augmentation strategies using backpropagation*, arXiv preprint arXiv:1911.06987, (2019).

[23] K. HE, X. ZHANG, S. REN, AND J. SUN, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[24] T. HERTZ, A. B. HILLEL, AND D. WEINSHALL, *Learning a kernel function for classification with small training samples*, in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 401–408.

[25] Y. HOSHEN AND J. MALIK, *Non-adversarial image synthesis with generative latent nearest neighbors*, CoRR, abs/1812.08985 (2018).

[26] H. HU, J. GU, Z. ZHANG, J. DAI, AND Y. WEI, *Relation networks for object detection*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3588–3597.

[27] S. IOFFE AND C. SZEGEDY, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, arXiv preprint arXiv:1502.03167, (2015).

[28] J. JOHNSON, A. ALAHI, AND L. FEI-FEI, *Perceptual losses for real-time style transfer and super-resolution*, in European conference on computer vision, Springer, 2016, pp. 694–711.

[29] A. KARPATHY, *t-sne visualization of cnn codes*, 2014.

[30] T. KARRAS, S. LAINE, AND T. AILA, *A style-based generator architecture for generative adversarial networks*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4401–4410.

[31] D. P. KINGMA AND M. WELLING, *Auto-encoding variational bayes*, 2013.

[32] S. KOBAYASHI, *Contextual augmentation: Data augmentation by words with paradigmatic relations*, in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), New Orleans, Louisiana, June 2018, Association for Computational Linguistics, pp. 452–457.

[33] A. KRIZHEVSKY, V. NAIR, AND G. HINTON, *Cifar-100 (canadian institute for advanced research)*, Canadian Institute for Advanced Research, (2009).

[34] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *Imagenet classification with deep convolutional neural networks*, in NIPS, 2012.

[35] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *Imagenet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds., Curran Associates, Inc., 2012, pp. 1097–1105.

[36] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *Imagenet classification with deep convolutional neural networks*, in Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, USA, 2012, Curran Associates Inc., pp. 1097–1105.

[37] J. LEHTINEN, J. MUNKBERG, J. HASSELGREN, S. LAINE, T. KARRAS, M. AITTALA, AND T. AILA, *Noise2noise: Learning image restoration without clean data*, arXiv preprint arXiv:1803.04189, (2018).

[38] J. LEMLEY, S. BAZRAFKAN, AND P. CORCORAN, *Smart augmentation learning an optimal data augmentation strategy*, Ieee Access, 5 (2017), pp. 5858–5869.

[39] W. LI, L. WANG, J. XU, J. HUO, Y. GAO, AND J. LUO, *Revisiting local descriptor based image-to-class measure for few-shot learning*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 7260–7268.

[40] S. LIM, I. KIM, T. KIM, C. KIM, AND S. KIM, *Fast autoaugment*, in NeurIPS, 2019.

[41] X. LIU, Y. ZOU, L. KONG, Z. DIAO, J. YAN, J. WANG, S. LI, P. JIA, AND J. YOU, *Data augmentation via latent space interpolation for image classification*, in 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 728–733.

[42] M. MIRZA AND S. OSINDERO, *Conditional generative adversarial nets*, CoRR, abs/1411.1784 (2014).

[43] A. NICHOL, J. ACHIAM, AND J. SCHULMAN, *On first-order meta-learning algorithms*, arXiv preprint arXiv:1803.02999, (2018).

[44] A. ODENA, *Semi-supervised learning with generative adversarial networks*, arXiv preprint arXiv:1606.01583, (2016).

[45] D. S. PARK, W. CHAN, Y. ZHANG, C.-C. CHIU, B. ZOPH, E. D. CUBUK, AND Q. V. LE, *Specaugment: A simple data augmentation method for automatic speech recognition*, arXiv preprint arXiv:1904.08779, (2019).

[46] A. RADFORD, L. METZ, AND S. CHINTALA, *Unsupervised representation learning with deep convolutional generative adversarial networks*, arXiv preprint arXiv:1511.06434, (2015).

[47] S. RIFAI, Y. N. DAUPHIN, P. VINCENT, Y. BENGIO, AND X. MULLER, *The manifold tangent classifier*, in Advances in Neural Information Processing Systems 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, eds., Curran Associates, Inc., 2011, pp. 2294–2302.

[48] O. RUSSAKOVSKY, J. DENG, H. SU, J. KRAUSE, S. SATHEESH, S. MA, Z. HUANG, A. KARPATHY, A. KHOSLA, M. S. BERNSTEIN, A. C. BERG, AND F. LI, *Imagenet large scale visual recognition challenge*, CoRR, abs/1409.0575 (2014).

[49] A. SANTORO, S. BARTUNOV, M. BOTVINICK, D. WIERSTRA, AND T. P. LILLICRAP, *One-shot learning with memory-augmented neural networks*, CoRR, abs/1605.06065 (2016).

[50] E. SCHWARTZ, L. KARLINSKY, J. SHTOK, S. HARARY, M. MARDER, A. KUMAR, R. FERIS, R. GIRYES, AND A. BRONSTEIN, *Delta-encoder: an effective sample synthesis method for few-shot object recognition*, in Advances in Neural Information Processing Systems, 2018, pp. 2845–2855.

[51] K. SHOEMAKE, *Animating rotation with quaternion curves*, SIGGRAPH Comput. Graph., 19 (1985), pp. 245–254.

[52] P. Y. SIMARD, D. STEINKRAUS, AND J. C. PLATT, *Best practices for convolutional neural networks applied to visual document analysis*, Seventh International

Conference on Document Analysis and Recognition, 2003. Proceedings., (2003), pp. 958–963.

[53] J. SNELL, K. SWERSKY, AND R. ZEMEL, *Prototypical networks for few-shot learning*, in Advances in Neural Information Processing Systems, 2017, pp. 4077–4087.

[54] M. A. TANNER AND W. H. WONG, *The calculation of posterior distributions by data augmentation*, in The Calculation of Posterior Distributions by Data Augmentation, 1987.

[55] S. THRUN AND L. Y. PRATT, *Learning to learn*, in Springer US, 1998.

[56] V. V. VAPNIK, *Statistical learning theory*, 1998.

[57] O. VINYALS, C. BLUNDELL, T. LILLICRAP, D. WIERSTRA, ET AL., *Matching networks for one shot learning*, in Advances in neural information processing systems, 2016, pp. 3630–3638.

[58] C. WAH, S. BRANSON, P. WELINDER, P. PERONA, AND S. BELONGIE, *The Caltech-UCSD Birds-200-2011 Dataset*, Tech. Rep. CNS-TR-2011-001, California Institute of Technology, 2011.

[59] X. WANG, H. PHAM, Z. DAI, AND G. NEUBIG, *SwitchOut: an efficient data augmentation algorithm for neural machine translation*, in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, Oct.-Nov. 2018, Association for Computational Linguistics, pp. 856–861.

[60] T. WHITE, *Sampling generative networks: Notes on a few effective techniques*, CoRR, abs/1609.04468 (2016).

[61] S. ZAGORUYKO AND N. KOMODAKIS, *Wide residual networks*, arXiv preprint arXiv:1605.07146, (2016).

[62] H. ZHANG, M. CISSE, Y. N. DAUPHIN, AND D. LOPEZ-PAZ, *mixup: Beyond empirical risk minimization*, arXiv preprint arXiv:1710.09412, (2017).

[63] X. ZHANG, Z. WANG, D. LIU, AND Q. LING, *Dada: Deep adversarial data augmentation for extremely low data regime classification*, in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 2807–2811.

[64] X. ZHANG, J. ZHAO, AND Y. LECUN, *Character-level convolutional networks for text classification*, in Advances in neural information processing systems, 2015, pp. 649–657.

[65] Z. ZHONG, L. ZHENG, G. KANG, S. LI, AND Y. YANG, *Random erasing data augmentation*, arXiv preprint arXiv:1708.04896, (2017).