Generative Latent Implicit Conditional Optimization when Learning from Small Sample

Idan Azuri
School of Computer Science and Engineering,
The Hebrew University, Jerusalem, Israel
idan.azuri@cs.huji.ac.il

Daphna Weinshall
School of Computer Science and Engineering,
The Hebrew University, Jerusalem, Israel
daphna@cs.huji.ac.il

***Abstract*—We revisit the long-standing problem of *learning from small sample*, to which end we propose a novel method called *GLICO* (Generative Latent Implicit Conditional Optimization). *GLICO* learns a mapping from the training examples to a latent space, and a generator that generates images from vectors in the latent space. Unlike most recent works, which rely on access to large amounts of unlabeled data, *GLICO* does not require access to any additional data other than the small set of labeled points. In fact, *GLICO* learns to synthesize completely new samples for every class using as little as 5 or 10 examples per class, with as few as 10 such classes without imposing any prior. *GLICO* is then used to augment the small training set while training a classifier on the small sample. To this end our proposed method samples the learned latent space using spherical interpolation, and generates new examples using the trained generator. Empirical results show that the new sampled set is diverse enough, leading to improvement in image classification in comparison with the state of the art, when trained on small samples obtained from CIFAR-10, CIFAR-100, and CUB-200.**

I. 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 question remains relevant to date, as collecting a very large set of images may be difficult due to issues of privacy, image quality, geographical location, time period, or copyright status. The difficulties are further aggravated in applications that require the acquisition and processing of a new custom dataset of images, which may be a highly costly task.

In the strict small sample settings, the learner has access to a small number of labeled examples from each class, possibly as few as 5 or 10, and the number of classes can also be rather small. This setting is substantially different from the two related settings of semi-supervised learning and the few-shots learning scenario. In the few-shot scenario, the learner has access to a large number of labeled examples from classes not participating in the current classification task, while in the semi-supervised scenario the learner typically has access to a large number of unlabeled examples. Thus most few-shot algorithms rely on transfer learning from tens of thousands of labeled training examples, while most semi-supervised algorithms transfer knowledge from the distribution of the unlabeled data.

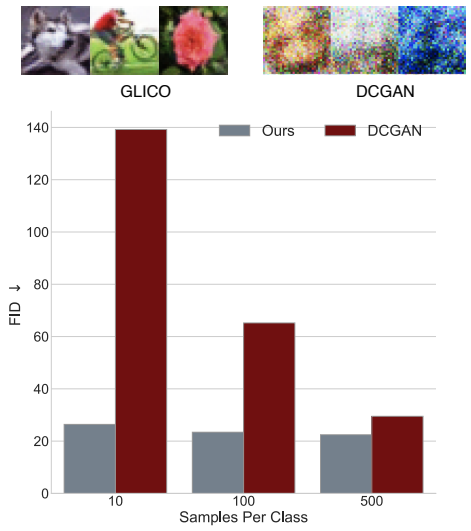


Fig. 1. The quality of image generation in a small sample settings. We compare our method to DCGAN with the same generator. The quality of the generated images is measured using the FID score [1] (lower is better). Top: new synthesized examples using our method (GLICO) and a widely used GAN model (DCGAN), both trained on CIFAR-100 with only 10 samples per class. Bottom: FID scores for each generator, when using 10, 100 and 500 examples from each class in CIFAR-100.

Methods designed to address the strict small sample scenario cannot rely on transfer learning from large amounts of peripheral data. Different approaches have been explored to address this problem, as reviewed in the next section. The problem can be notably alleviated by imposing a strong prior on the model. Unfortunately, in many applications we face an unknown domain and such prior knowledge is not available.

Data augmentation may help, reflecting the availability of some weak prior knowledge about the data. Methods employing semi-supervised [2] and transductive learning [3] 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 in conjunction to further boost performance as shown in Section V-C.

Why not GAN? Using generative models to augment a small 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. These models perform poorly in the strict small sample scenario, as can be seen in Fig. 1. To obviate this problem, our method optimizes the latent space directly, similarly to [4]. It is thus able to effectively synthesize quality new images in the small sample domain, outperforming the conditional DCGAN as can be seen in Fig. 1, where image quality is measured using the FID score.

More specifically, our proposed method (see Fig. 2) learns a latent space code for each data point separately. To improve the properties of the learned latent space, we seek to push intra-class examples to lie close to each other and separated from inter-class examples as much as possible. This is accomplished by adding a classifier to the basic architecture as described in Section III-B. The classifier 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. The full model is described in Section III.

The architecture of the model reveals geometric properties that allow us to effectively sample specific areas in the latent space and synthesize novel images as explained in Section III-B. The empirical evaluation described in Section IV 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 paper is organized as follows. In Section III we describe *GLICO*, a novel method for data synthesis which can be effectively trained from a few labeled points from each class. In Section IV we describe how synthetic images are used to boost the training of a discriminative deep classifier. In Section V we demonstrate the superior performance of our method when compared to alternative methods under extreme low sample conditions in the strict small sample scenario with no additional unlabeled data. We provide an ablation study and further investigate alternative design choices and their effect on classification performance. In Section V-C we show empirically that the synthetic examples produced by *GLICO* are unlike those generated by conventional augmentation methods. Lastly, we investigate the contribution of adding unlabelled data to *GLICO* in the semi supervised settings in Section V-D.

II. Related Work

Earlier work addressed the small sample scenario mainly in the context of Bayesian inference, see for example [5]–[7]. These methods have been revisited in recent years in the context of *few-shot learning*, where transfer learning can be exploited. 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.

Data Augmentation with Geometrical Transformations. Focusing our attention back at the small sample problem, *data augmentation* as discussed in [8] can be effectively used. Common augmentation techniques for image classification include translating the image by a few pixels, adding Gaussian noise, and flipping the image horizontally or vertically [9]–[11]. Additional image augmentation methods include *Cutout* [12], 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* [13], where similarly to dropout randomly chosen rectangular regions in the image have their pixels erased or replaced by random values. *MixUp* [14] 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* [15] 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 I.

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, see for example [14], [16]–[20]. These methods typically require large amounts of (labeled or unlabeled) data to be effective. They try to leverage generative models by sampling new synthetic examples from the learned distribution of the data. Unfortunately in the small sample case, when transfer learning is not an option, the few labeled examples do not represent the true data distribution very reliably, resulting in poor generalization and low-quality synthetic data. Finally, the notorious instability of their training process and the heavy computational load make GANs less appealing for mere data augmentation.

III. Our Model

We propose a method for multi-class image classification from small sample, which consists of data augmentation with a generative model. Specifically, 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 is a latent optimization method that combines an auxiliary task of classification, and which can be trained effectively from a small sample. The architecture is designed 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].

A. Model Overview

The generative model we use to sample new data includes a basic generative latent optimization architecture with generator

[1]https://github.com/IdanAzuri/glico-learning-small-sample

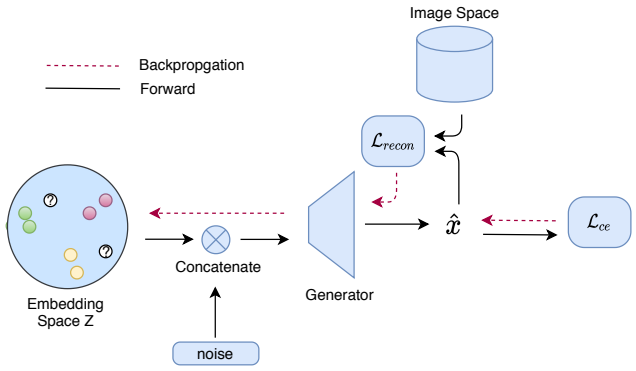


Fig. 2. Schematic illustration of our method. Every $\mathbf{z}_i \in Z$, where Z denotes the unit sphere, is mapped to a specific image x_i in the image space. In this illustration, the color of each image frame marks the class label of the image, while black frames mark unlabeled images. The classifier is used to propagate error only when a label is given.

G_θ, and an added small classifier f_ϕ which is trained to classify the labeled data (see Fig. 2). Training is not adversarial, and therefore this classifier is nothing like the discriminator in the GAN architecture. The latent space is initialized randomly $\{\mathbf{z}_i \in Z\}$ where Z denotes 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 (2) is used to train G_θ as done in [4]. 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. 2).

Here we use the perceptual loss [21] to measure the reconstruction loss. More specifically, in order to compute the perceptual loss we extract the activation vectors in layers '$conv_{1\_2}$', '$conv_{2\_2}$', '$conv_{3\_2}$', '$conv_{4\_2}$' 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:

$$\mathcal{L}_{percep}(x_i, \mathbf{z}_i; \theta) = \sum_j \lambda_j ||\xi_j(G_\theta([\mathbf{z}_i, \varepsilon])) - \xi_j(x_i))||_1 \quad (1)$$

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

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

B. Generative Latent Optimization

Generative model. Generative Latent Optimization (GLO) is an effective and thin method for image reconstruction, relying on a 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 $\{x_1, x_2 \dots x_n\} \in X$ 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 $\{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\} \in Z$ where $Z \subseteq \mathbb{R}^d$. Pair

Algorithm 1 GLICO. The algorithm learns codes $\{\mathbf{z}_i\}_{i=1}^n$ by minimizing the reconstruction loss $\mathcal{L}_{percep}$ of generator G_θ, and the cross entropy loss $\mathcal{L}_{ce}$ of discriminator f_ϕ.

Input: unlabeled data P_{D_u}, labeled data P_{D_l}, γ, epochs
$n = |P_{D_l}| + |P_{D_u}|$
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**
    $\varepsilon \sim N(0, \sigma I)$
    $\mathcal{L}_{percep} = \sum_j \lambda_j ||\xi_j(G_\theta([z_i, \varepsilon])) - \xi_j(x_i))||_1$
    $\mathcal{L}_{ce} = \mathcal{L}_{CE}(f_\phi(G_\theta([z_i, \varepsilon])), y_i)$
    $\mathcal{L} = \mathcal{L}_{percep} + \gamma \mathcal{L}_{ce}$
    Update $\{\mathbf{z}_i\}, \theta, \phi$ using the gradient of $\mathcal{L}$
    $\forall \mathbf{z}_i \in Z$, $\mathbf{z}_i := \frac{\mathbf{z}_i}{||\mathbf{z}_i||_2}$
  end for
  // Optional: transductive learning mode
  for $x_i \in P_{D_u}$ **do**
    $\mathcal{L} = \sum_j \lambda_j ||\xi_j(G_\theta([z_i, \varepsilon])) - \xi_j(x_i))||_1$
    Update $\{\mathbf{z}_i\}, \theta$ using the gradient of $\mathcal{L}$
    $\forall \mathbf{z}_i \in Z$, $\mathbf{z}_i := \frac{\mathbf{z}_i}{||\mathbf{z}_i||_2}$
  end for
  $epoch += 1$
until $epoch > epochs$

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

$$\min_\theta \sum_{i=1}^n \left[\min_{\mathbf{z}_i \in Z} \mathcal{L}_{percep}(x_i, \mathbf{z}_i; \theta) \right] \quad s.t. \quad ||\mathbf{z}_i||_2 = 1 \quad (2)$$

While in GLO the Laplacian pyramid is used for reconstruction loss, we note that the minimization of $\mathcal{L}_{percep}$ appears to yield more realistic results [22], as compared to other image metrics in common use.

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. To decrease the intra-class distances and increase the inter-class distances in the latent space representation, we propose the conditional model termed *GLICO*. 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, $\mathcal{L}_{percep}$ in (2) is replaced by $\mathcal{L}_{percep} + \mathcal{L}_{ce}$, where $\mathcal{L}_{ce}$ is the cross-entropy loss of classifier f_ϕ. Fig. 3 illustrates the structure of the latent space when the different loss functions are used.

Sampling the Latent Space. Generating images based on randomly sampling the latent space, even when restricted to the immediate vicinity of $\{\mathbf{z}_i\}_{i=1}^n$, produces low quality somewhat

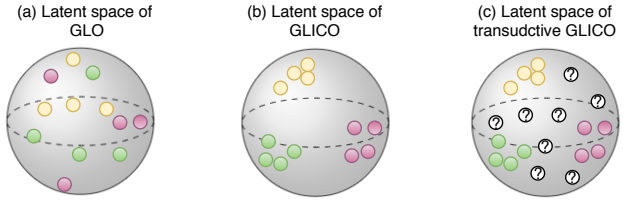


Fig. 3. Illustration of the latent space Z. (a) GLO: vectors $\mathbf{z}_i \in Z$ do not have semantic meaning in Z. (b) Our method: vectors from the same class are grouped. (c) Our method in transductive mode. Notations: filled colored circles represent different labeled datapoints, where color corresponds to class identity. Black circles with the symbol "?" represent unlabeled datapoints.

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*) [23], which is defined as follows:

$$slerp(q1, q2; t) = q_1 \frac{\sin(1-t)\vartheta}{\sin\vartheta} + q_2 \frac{\sin t\vartheta}{\sin\vartheta} \qquad (3)$$

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

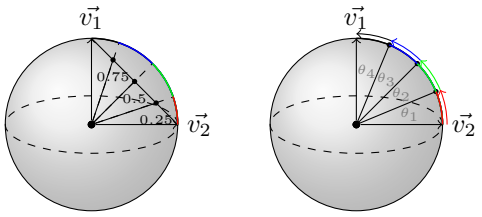


Fig. 4. *Slerp* vs. *lerp*. Left side: linear interpolation between $\vec{v_1}$ and $\vec{v_2}$ with $t \in [0.25, 0.5, 0.75]$. Right side: spherical interpolation. Note that both the length and arc length of the interpolated vectors are equal in *slerp* but unequal in *lerp*.

As shown in Fig. 4, the *slerp* interpolation follows the great circle path on a 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, with both uniform and Gaussian priors [24].

Why *slerp*? Linear interpolation (*lerp*) is the simplest method to traverse the latent space manifold between two known locations. Often it is used to illustrate learned features that capture the semantics of the dataset [25]. However, [26] 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 $[\mathbf{z}_i, \varepsilon]$, $\varepsilon \sim N(0, \sigma I)$, $\mathbf{z}_i \in \{\mathbf{z} \in Z : ||\mathbf{z}||_2 = 1\}$), see Fig. 2.

Transductive learning. The setting of transductive learning was introduced by [3]. Like semi-supervised learning it aims to exploit unlabeled data, using the unlabeled test set to improve the test set classification.

Specifically, the learning task is defined on a given set of ℓ training points

$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_\ell, y_\ell)\}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$$

and a sequence of k test vectors $\{\mathbf{x}_{\ell+1}, \ldots, \mathbf{x}_{\ell+k}\}$. The goal of the learner is to find among an admissible set of binary vectors the one that classifies the test vectors as accurately as possible. We assume that the set of vectors $\{\mathbf{x}_{\ell+1}, \ldots, \mathbf{x}_{\ell+k}\}$ and their corresponding true labels is an i.i.d sample drawn according to the same unknown distribution $P(\mathbf{x})$ and $P(y|\mathbf{x})$. Basically, the core idea in transductive learning is to leverage the implicit information in the instances whose output is required $(\mathbf{x}_{\ell+1}, \ldots, \mathbf{x}_{\ell+k})$, in order to improve their classification.

In the same spirit as the classical setting of transductive learning, we may use the unsupervised data to train the generator of the model to reconstruct the test points while optimizing their reconstruction loss (2), see Fig. 3.

IV. Evaluation Methodology

A. Datastes

We evaluate our method on three standard benchmarks for image classification. The first two are CIFAR-10 and CIFAR-100 [27], each includes $50,000$ 32×32 color images, with 10 or 100 classes and 500 or 5000 images per class respectively. The relatively small size of the images allows us to perform an exhaustive ablation study on this dataset as described in Section V. The second dataset is CUB-200 [28], 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.

B. 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 [29]. As classification engine we used, unless otherwise noted, *WideResNet-28* [30] for CIFAR-10 and CIFAR-100, and Resnet50 [31] 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 *GLICO*. 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 probability 0.5. Otherwise (with probability 0.5) it is replaced by a new image obtained by sampling the latent space $G([slerp(\mathbf{z}_i, \mathbf{z}_j, t), \varepsilon]))$. $\mathbf{z}_j$ 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 *GLICO* to image augmentation with Cutout [12], Random Erase [13] and MixMatch [15]. In each case we repeated the same procedure as described above, replacing the generation of a new image using *GLICO* by an image obtained from the corresponding augmented set of images. We also evaluated the method described in [29], which was explicitly designed to handle small sample, using code provided by the authors.

C. Implementation Details

We used SGD optimization with learning rate 0.1 for CIFAR-10 and 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, see Section V-E.) 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.

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 [32]. The generator of any 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 the generator of a relatively simple GAN architecture. We trained the model on 2 $\times$ RTX-2080 GPUs for 200 epochs; every epoch took around 40 seconds on CIFAR-100 and 3 minutes on CUB-200.

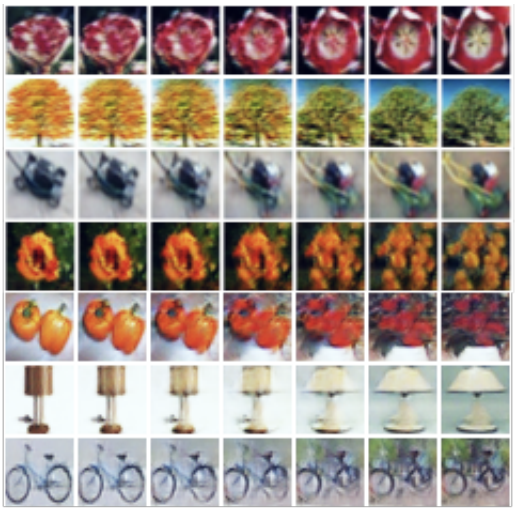
To achieve uniformity we oversampled the training set in proportion to the size of the training set. For example, with 50 samples per class in CIFAR-100, we trained the model $\times 10$ iterations. Standard errors (STE) were obtained from 3 runs with different seeds in all study cases except for MixMatch,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

V. Results

The results of our empirical study are summarized in Fig. 6 for CIFAR-10, and Table I for CIFAR-100 and CUB-200. We used small sample partitions, with 10 to 250 labeled samples per class (SPC) in CIFAR-10, 10 to 100 labeled SPC in CIFAR-100, and 5 to 30 labeled SPC in CUB-200. We compared our model to three different methods of data augmentation and one small sample method.

Fig. 5a 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. Fig. 5b Illustrates the quality of the reconstruction when using *GLICO* with 10 samples per class only, demonstrating high quality reconstruction.

The two augmentation methods used for comparison in our experiments are Random Erasing [13] and Cutout [12]. In all



(a) CIFAR-100 image generation.

(b) CUB-200 image reconstruction.

Fig. 5. Examples of synthesized images. (a) Each row shows five new images (the intermediate columns), generated based on smooth interpolation in the latent space between two reconstructed images (the left and right columns). (b) High-resolution image reconstruction. Here *GLICO* was trained only on *10 examples* per class from CUB-200.

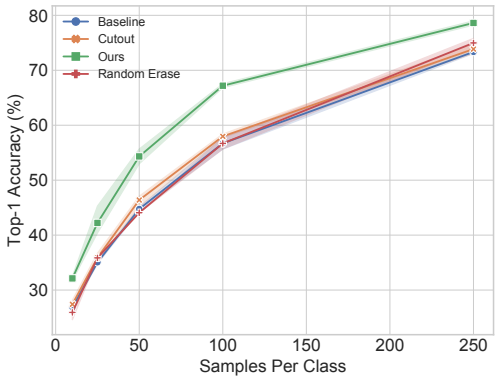


Fig. 6. Comparison of Top-1 Accuracy (including STE) for CIFAR-10 using WideResnet-28, with a different number of training examples per class (labeled data only).

the study cases but one, augmentation with *GLICO* significantly outperforms the other methods (see Table I). With CIFAR-100 and 100 samples per class, Cutout matches the highest accuracy (similar to *GLICO*) in our experiments.

MixMatch [15] 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 Table I, but see Section V-D. Possibly, the blending of images in pixel space, which is intended to provide some means of regularization, is only effective when enough training data are available. Otherwise, it feeds noisy examples to the model and makes it harder to

TABLE I

COMPARISON OF TOP-1 ACCURACY (INCLUDING STE) FOR CIFAR-100 AND CUB-200 USING WIDERESNET-28 AND RESNET-50 RESPECTIVELY, WITH A DIFFERENT NUMBER OF TRAINING EXAMPLES PER CLASS (LABELED DATA ONLY). THE METHODS USED FOR COMPARISON ARE DESCRIBED IN THE TEXT BELOW. BEST RESULTS ARE MARKED IN BOLD. FOR [29], * INDICATES THAT THE REPORTED RESULTS, AS OBTAINED IN OUR EXPERIMENTS USING CODE RELEASED BY THE AUTHORS, DO NOT MATCH THE RESULTS REPORTED BY THE AUTHORS WHICH ARE THEREFORE LISTED IN PARENTHESES.

DATASET	SAMPLES/CLASS	BASELINE	OURS	MIXMATCH	CUTOUT	RANDOM ERASE	[29]*
CIFAR-100	10	22.89±0.09	**28.55±0.40**	24.80	23.43±0.24	23.26±0.27	23.01 (22)
	25	38.39±0.10	**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)
CUB-200	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)

generalize.

[29] 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 Table I. Admittedly, we were not able to reproduce their published results, which are therefore noted in parentheses[2].

A. Ablation Study

Next, we review and evaluate different design choices used in the architecture and the approach proposed in this paper. The results are summarized in Table II.

TABLE II

ABLATION STUDY: TRAINED ON CIFAR-100 WITH 25 LABELED TRAINING EXAMPLES PER CLASS. TOP 1 AND TOP 5 ACCURACY IS CALCULATED BASED ON THE ARCHITECTURAL VARIANTS DESCRIBED SECTION V-A.

Model	Top 1 Acc.	Top 5 Acc.
GLICO	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	44.79±0.12	71.28±0.09

More specifically, we see in Table II the effect of omitting different components of *GLICO*, 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. 'Transductive" shows the added benefit obtained from including the unlabeled test set in the training of generator G_θ.

B. Additional Design Choices

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

[2]The original published 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.

TABLE III

TOP 1 AND TOP 5 ACCURACY OF ADDITIONAL DESIGN CHOICES AS EXPLAINED IN THE TEXT.

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

Latent Classifier. One can optimize the discriminative loss $\mathcal{L}_{CE}$ 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.

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

Additive Noise. *GLICO* relies on the concatenation of noise to the latent space representation. To investigate the contribution of this component, and following [33], we 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 (2), so that $\hat{x} = G(z_i + \varepsilon)$. 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 III, this approach leads to performance degradation in the final classification.

Cosine Loss. It is argued in [29] that the cosine loss provides a better optimization function for the small sample regime. In our experimental setup, the cross-entropy classification loss provided better results, see Table III.

C. Relation to Classical Augmentations

So far we have shown that our method boosts classification performance in the small sample settings when augmenting the small training set with images synthesized by our generative model. But are we learning to generate any significant new

information? In other words, can similar images and the same boost in performance be obtained by simpler alternative means of image augmentation?

We approach this question by reevaluating the results of our method, modified so that new images are synthesized by an alternative image augmentation technique which employs classical geometric transformation. To make the challenge as hard as possible, we adopt AutoAugment (AA) [11], an RL based augmentation method, which estimates the optimal set of classical transformations to augment images in CIFAR-100. AA exhaustively searches through 16 types of color-based and geometric base transformations, while using *all the images* in CIFAR-100 benchmark dataset. Note that this gives an unfair advantage to this method as compared to our original method. The case studied here is CIFAR-100 with 50 labeled samples per class, and with transductive learning (similar results are obtained without transductive learning).

TABLE IV
TOP-1 AND TOP-5 ACCURACY WHEN AUGMENTING A SMALL DATASET (CIFAR-100, 50 SPC) BY *GLICO* 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. THE FIRST ROW PROVIDES THE BASELINE.

AutoAu.	Ours	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

The results of this challenge are shown in Table IV. Clearly each method boosts classification performance, as can be seen in rows 2-3 when compared to the baseline in row 1. But when using the two methods - AA and *GLICO* - together, performance improves even further (row 4). It appears that each augmentation method provides an independent contribution, and that the effect of the two augmentation methods is additive. From this empirical result we conclude that the contribution of *GLICO* goes beyond the contribution of augmentation by classical image transformations.

D. Using Unlabeled Data

While our method is designed to address the strict small sample settings, the same approach can be beneficial in the semi-supervised settings (SSL), where the learner is given access in addition to a large set of unlabeled images.

Note that since *GLICO* is not designed to resolve the SSL problem, it does not have any mechanism of 'label guessing' like other SSL methods. The following analysis simply aims to explore the limits of the current model in different settings. Thus, while in the fully supervised scenario *GLICO* outperforms MixMatch as shown in Table I, MixMatch slowly improves when given access to unlabeled data, and eventually outperforms *GLICO* as shown in Table V. Clearly MixMatch benefits from unlabeled data more considerably. In order to be effective in the semi-supervised settings, we will need to enhance *GLICO* with some mechanism of label guessing.

TABLE V
SEMI-SUPERVISED SCENARIO: TOP-1 ACCURACY OF MIXMATCH VS GLICO WHEN SHOWN CIFAR-100 WITH 25 LABELED EXAMPLES PER CLASS AND A VARYING NUMBER OF UNLABELED EXAMPLES, WHERE EACH CASE CORRESPONDS TO A DIFFERENT COLUMN.

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

E. Weak Vs. Strong Classifier

Our algorithm is designed to optimize a combined loss $\mathcal{L}_{percep} + \mathcal{L}_{ce}$, where $\mathcal{L}_{percep}$ drives the generator to achieve good reconstruction, while $\mathcal{L}_{ce}$ serves as a regularizer. $\mathcal{L}_{ce}$ imposes structure on the latent space reflecting the known labels. The minimization of the regularizer $\mathcal{L}_{ce}$ is mediated by a classifier f_ϕ, while the minimization of the reconstruction loss $\mathcal{L}_{percep}$ is mediated by a generator G_θ. A strong classifier f_ϕ, e.g. Resnet-50 or VGG-19 with 45M and 144M parameters respectively, has more than x5 parameters as compared to off-the-shelf generators G_θ. This would shift the balance of the learning algorithm from the generative to the discriminative component of the algorithm. Thus it would seem that the strength of the classifier f_ϕ should be controlled to reflect the amount of labeled data, or how small the sample is.

In Fig. 7 we evaluate four classifiers: 3 strong classifiers including Resnet-50, VGG-19 and Wide-Resnet28, and one weak classifier which is a small CNN with 4 convolutional layers. Our results show that the smaller model achieves the best results in the low regime of the small sample settings, while VGG19 achieves higher accuracy when more labeled data are available.

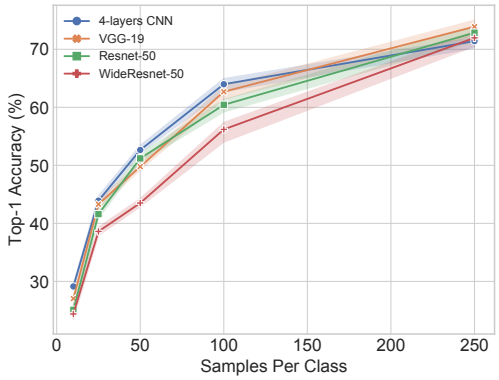


Fig. 7. Weak vs. strong classifier. Top-1 Accuracy (including STE) for CIFAR-10 when using different architectures for the classifier component in our method. In the small regime (10-100 samples per class) the weak classifier outperforms the stronger classifiers. When sufficient labelled examples are available, the stronger classifiers achieve higher accuracy.

VI. SUMMARY AND DISCUSSION

In this work we revisited the problem of learning from small sample. We developed a deep generative model, called Generative latent implicit conditional optimization (*GLICO*),

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-10, CIFAR-100, and CUB-200 benchmarks, 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 to achieve decent reconstruction results as demonstrated in Fig. 5.

The optimization of each representation vector 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 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 $\mathcal{L}_{CE}$ 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).

The unique aforementioned properties of our model allow it to improve the training efficacy of deep classifiers in the small sample regime. 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.

ACKNOWLEDGEMENTS

This work was supported by a grant from the Israel Science Foundation (ISF), a grant from the Israel Innovation Authority (Phenomics), and by the Gatsby Charitable Foundations.

REFERENCES

[1] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a nash equilibrium," *CoRR*, vol. abs/1706.08500, 2017. [Online]. Available: http://arxiv.org/abs/1706.08500

[2] A. Odena, "Semi-supervised learning with generative adversarial networks," *arXiv preprint arXiv:1606.01583*, 2016.

[3] V. V. Vapnik, "Statistical learning theory," 1998.

[4] 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*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholm, Sweden: PMLR, 10–15 Jul 2018, pp. 600–609. [Online]. Available: http://proceedings.mlr.press/v80/bojanowski18a.html

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

[6] S. Thrun and L. Y. Pratt, "Learning to learn," in *Springer US*, 1998.

[7] 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 *CVPR*, 2004.

[8] M. A. Tanner and W. H. Wong, "The calculation of posterior distributions by data augmentation," in *proceeding*, 1987.

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

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[11] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data," *CoRR*, vol. abs/1805.09501, 2018.

[12] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[13] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *arXiv preprint arXiv:1708.04896*, 2017.

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] T. DeVries and G. W. Taylor, "Dataset augmentation in feature space," *arXiv preprint arXiv:1702.05538*, 2017.

[20] 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.

[21] 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.

[22] Y. Hoshen and J. Malik, "Non-adversarial image synthesis with generative latent nearest neighbors," *CoRR*, vol. abs/1812.08985, 2018. [Online]. Available: http://arxiv.org/abs/1812.08985

[23] K. Shoemake, "Animating rotation with quaternion curves," *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 245–254, Jul. 1985. [Online]. Available: http://doi.acm.org/10.1145/325165.325242

[24] T. White, "Sampling generative networks: Notes on a few effective techniques," *CoRR*, vol. abs/1609.04468, 2016. [Online]. Available: http://arxiv.org/abs/1609.04468

[25] 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.

[26] G. Arvanitidis, L. K. Hansen, and S. Hauberg, "Latent space oddity: on the curvature of deep generative models," 2017.

[27] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-100 (canadian institute for advanced research)," *Dataset*, 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[28] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[29] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," *CoRR*, vol. abs/1901.09054, 2019. [Online]. Available: http://arxiv.org/abs/1901.09054

[30] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[31] 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.

[32] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[33] 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.