Generative Latent Implicit Conditional Optimization (GLICO) when Learning from Small Sample

Idan Azuri *Daphna Weinshall*

School of Computer Science and Engineering, The Hebrew University of Jerusalem

25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
Milan, Italy 10 | 15 January 2021

Full Paper Code

Background & Motivation

- **Small Sample Learning:**
  Learning from limited amounts of data (possibly as few as 5 or 10 samples per class) without any additional assumptions or prior knowledge.
- **Challenge:**
  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.
- **Key Idea:**
  - Learn image representation independently while keeping semantic structure of the latent space representation.
  - Achieve data enrichment using conditioned image generation by sampling the area around the learnt vector.
- **Our Solution:**
  - A novel generative model, GLICO, which learns from very small datasets, without using or imposing any prior.
  - Then, GLICO is used as an effective and versatile data augmentation method in the low data regime.
- **Innovation:**
  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.

Small Sample Learning ≠ Few Shot Learning

- The **Small Sample** settings are substantially **different** from the two related settings of **Semi-Supervised Learning** (SSL) and the **Few-Shot** (FS) **learning**.
- **FS-** The learner has access to many labeled examples from classes not participating in the current classification task. Thus, most FS algorithms rely on transfer learning from tens of thousands of labeled training examples.
- **SSL-** The learner typically has access to many unlabeled examples. Most SSL algorithms transfer knowledge from the distribution of the unlabeled data.

GLICO Training

- Let $\{x_i\}_{i=1}^n$ denote a set of labeled images, choose n d-dimensional random **learnable** vectors on the unit sphere $\{z_i\}_{i=1}^n$, $Z \subseteq \mathcal{R}^d$.
- Pair every image x_i with a random vector z_i, to achieve the mapping $\{(x_1, z_1), \dots, (x_n, z_n)\}$. Learn the parameters θ of the generator G and the optimal set $\{z_i\}$ by minimizing the objective:

$$\min_{G_\theta, \{z_i\},} \mathcal{L}_{recon}(G([z_i, \varepsilon]), x_i)$$

$$\text{s.t } \|z_i\| = 1, \varepsilon \in \mathcal{N}(0, \sigma I)$$

$\mathcal{L}_{recon}$ is the reconstruction loss, here we use perceptual loss with VGG-16.

- Jointly, learn the classifier $\mathcal{F}_\theta$ to classify the labeled data (x_i, y_i) when available:

$$\min_{\mathcal{F}_\phi} \mathcal{L}_{CE}\left(\mathcal{F}_\phi(G([z_i, \varepsilon])), y_i\right)$$

$\mathcal{L}_{CE}$ is the cross-entropy loss.

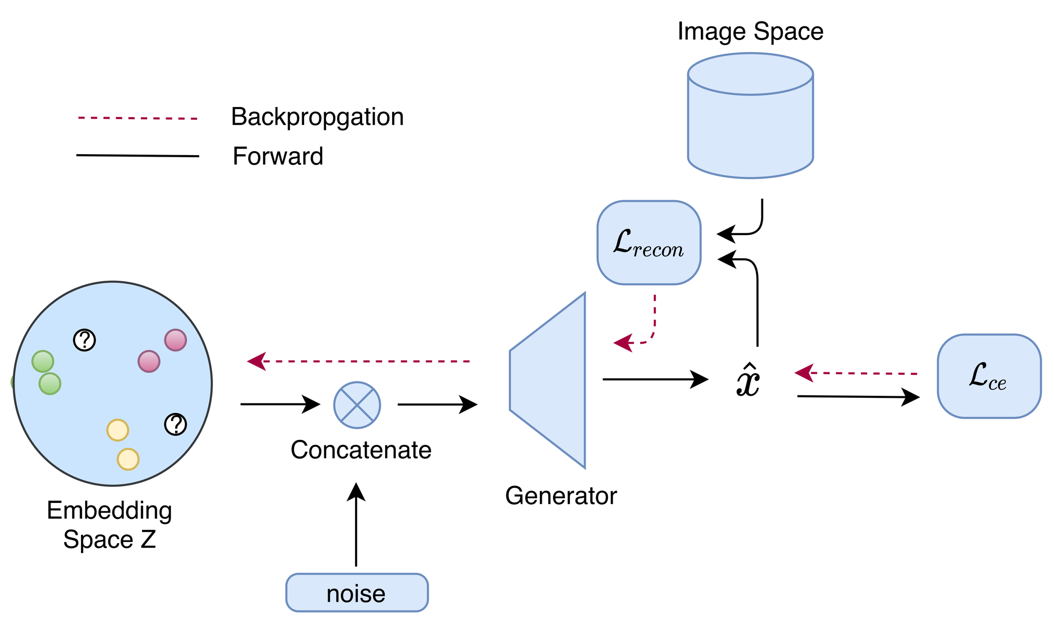


Image Classification Using GLICO

- Train the proposed model GLICO as described above by minimizing the sum of the reconstruction loss and the cross-entropy loss:

$$\min_{G_\theta, \{z_i\}, \mathcal{F}_\phi} \mathcal{L}_{recon} + \mathcal{L}_{CE}$$

- Train a classifier as follows:
  - Sample two pairs $(x, z)_i, (x, z)_j \in (Z, X)$
    $where\ x_i, x_j \in C_k,\ C_k \subset C, C\ is\ the\ classes\ set$
  - $z_{inter} = SLERP^*(z_i, z_j, r)$, $r \sim U[0, 0.4]$
  - Alternate with probability $\mathcal{P} = 0.5$ training inputs for the classifier $\hat{x}_i = G([z_{inter}, \varepsilon])$ and the image x_i

*Spherical linear interpolation

Conceptual Illustration

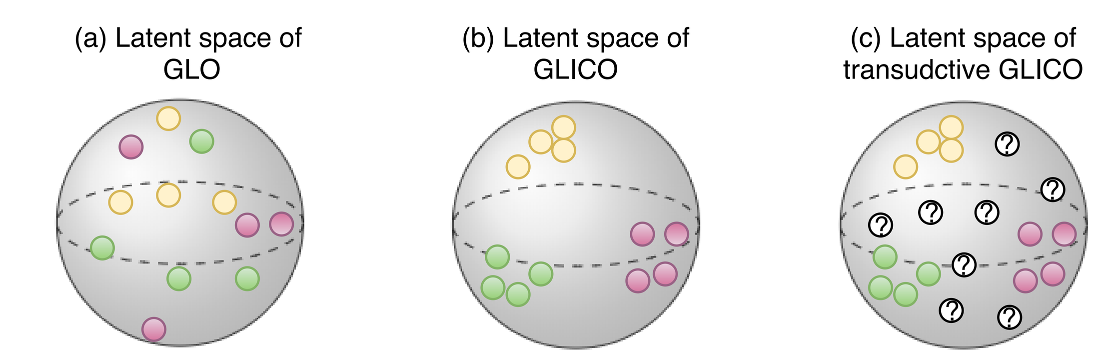


Illustration of the latent space Z.
(a) Naïve **Generative Latent Optimization (GLO):** Vectors $z_i \in Z$ do not have semantic meaning in Z space.
(b) **Our method:**
Vectors from the same class are grouped.
(c) **Our method in transductive mode.**

Notations: Filled colored circles represent different labeled datapoints, where color corresponds to class identity. Black circles with the symbol "?" represent unlabeled datapoints.

Experimental Results

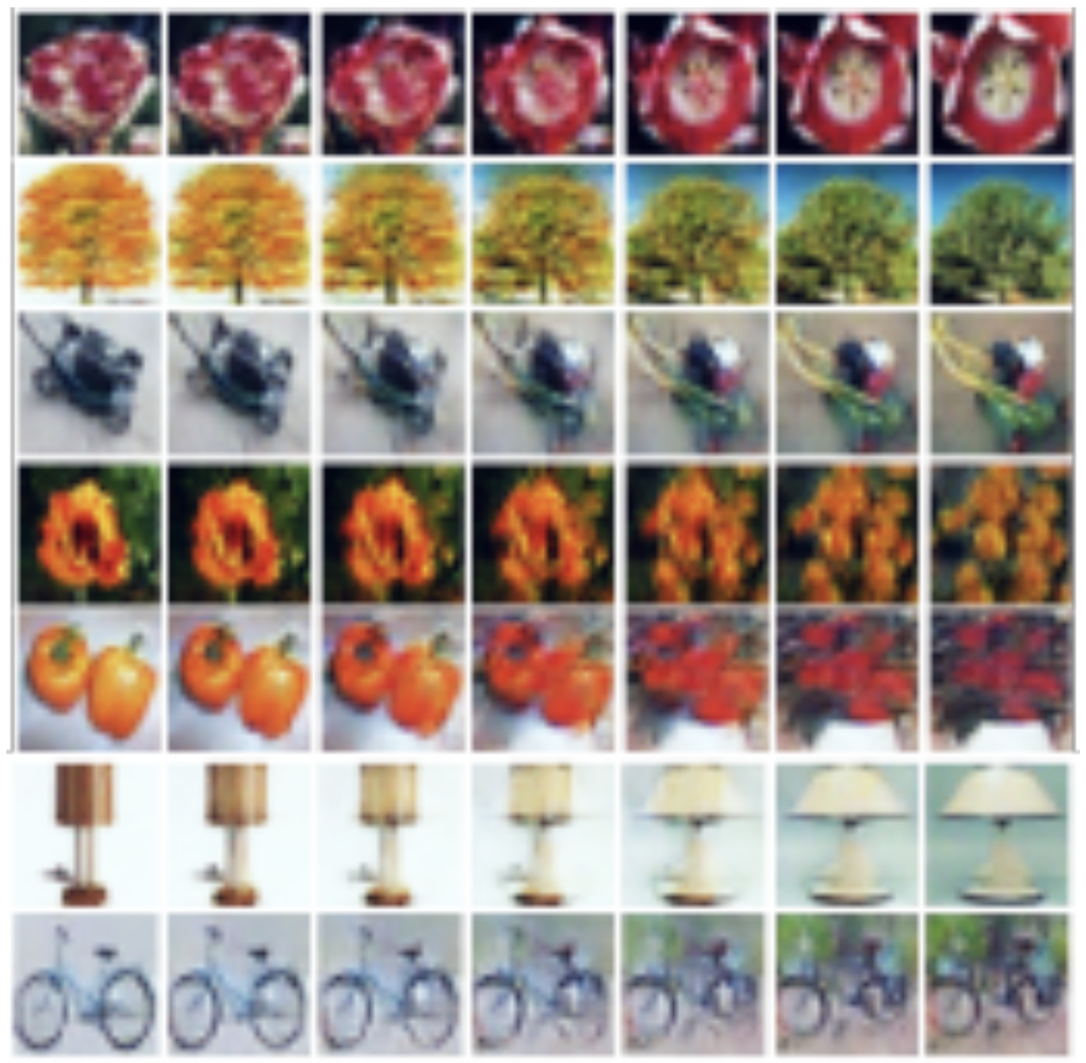
- For each benchmark, we defined a small sample task by subsampling the original training set of the corresponding dataset.
- Classification NN backbone are WideResNet-28 for CIFAR-10 and CIFAR-100, and Resnet50 for CUB-200.

Table: Top-1 accuracy (%) including STE with a different number of training samples per class (labeled data only).

Dataset	Samples/ Class	Baseline	Ours	MixMatch	Cutout	Random Erase	[1]*
CIFAR -100	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±0.10	**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)

* 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.

Examples of synthesized images



Are We Synthesizing Trivial Samples?

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. We adopt AutoAugment [2], an RL based augmentation using all the images in CIFAR-100 benchmark dataset.
The case studied here is CIFAR-100 with 50 labeled samples per class, and with transductive learning (similar results are obtained without transductive learning).
Clearly each method boosts classification performance, but when using the two methods – AutoAu. and GLICO - together, performance improves even further (row 4).

Table: Top-1 and Top-5 accuracy (%) when augmenting a small dataset (CIFAR-100, 50 samples per class), by GLICO alone (second row), Auto Augment alone (third row), or both (fourth row).

AutoAu. [2]	Ours	Top-1 Acc.	Top-5 Acc.
		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**

References

[1] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss", 2020
[2] E. D. Cubuk, B. Zoph, D. Man´e, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data", 2019