Few Shot Image Generation via Implicit Autoencoding of Support Sets

Shenyang Huang
Mila, McGill University
shenyang.huang@mail.mcgill.ca

Kuan-Chieh Wang
Vector Institute, University of Toronto
wangkua1@cs.toronto.edu

Guillaume Rabusseau
Mila, DIRO, Université de Montréal
guillaume.rabusseau@umontreal.ca

Alireza Makhzani
Vector Institute, University of Toronto
makhzani@vectorinstitute.ai

Abstract

Recent generative models such as generative adversarial networks have achieved remarkable success in generating realistic images, but they require large training datasets and computational resources. The goal of few-shot image generation is to learn the distribution of a new dataset from only a handful of examples by transferring knowledge learned from structurally similar datasets. Towards achieving this goal, we propose the "Implicit Support Set Autoencoder" (ISSA) that adversarially learns the relationship across datasets using an unsupervised dataset representation, while the distribution of each individual dataset is learned using implicit distributions. Given a few examples from a new dataset, ISSA can generate new samples by inferring the representation of the underlying distribution using a single forward pass. We showcase significant gains from our method on generating high quality and diverse images for unseen classes in the Omniglot and CelebA datasets in few-shot image generation settings.

1 Introduction

Deep generative models such as Variational Autoencoders (VAEs) [13] and Generative Adversarial Networks (GANs) [6] have shown significant improvement on image generation tasks. The idea of GANs relies on a two player min-max game where the generator learns a mapping from the noise vector to the image space and the discriminator is trained to differentiate samples from the data distribution and the learned sample distribution. The goal is to train the generator to produce image samples indistinguishable from samples of the real distribution. GANs are widely known for their ability to generate high-quality images [23, 11]. This can be attributed to the fact that GANs are capable of learning flexible implicit distributions, as opposed to prescribed density models [20, 9].

Successful learning of deep generative models typically requires a vast amount of training data. For example, 300M images across 18K categories were used in [2] to train GANs on natural images. In many practical settings, we are interested in learning the underlying distribution of a dataset from only a handful of examples. Few-shot image generation methods attempt to bridge this gap by exploiting the structural similarity between different datasets, and thereby significantly reducing the number of examples required for learning the distribution of a new dataset.

In this work, we propose a few-shot image generation method called "Implicit Support Set Autoencoder" (ISSA). ISSA is an adversarial framework that learns the structural similarities across datasets with an unsupervised context representation, while the distribution of each individual dataset is learned using implicit distributions. We use the term "support set", which is commonly used in

5th Workshop on Meta-Learning at NeurIPS 2021, held virtually.

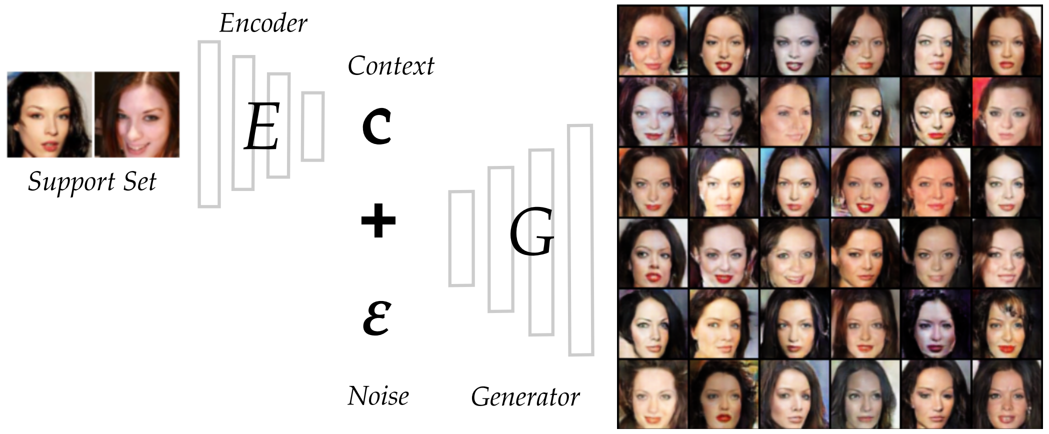


Figure 1: ISSA can generate diverse and high quality images of an unseen identity with as few as two support examples.

the few-shot classification literature, to refer to the few examples given from the new dataset. At test time, to learn the distribution of a novel dataset, we pass the given support set through the learnt encoder E to infer the context representation $\mathbf{c}$ for the dataset. Then the generator G conditions on this representation to generate new images from the underlying distribution of this dataset using the implicit generator. Figure 1 shows the inference process of our model on the CelebA faces dataset [16], where the new dataset consists of two images of a new identity which is not observed during training. Here, the encoder infers context $\mathbf{c}$ which is the representation for the unseen identity. The generator then conditions on the context vector and uses the stochasticity of the noise $\boldsymbol{\epsilon}$ to generate new images of the same identity but with different background, head pose, lighting, etc.

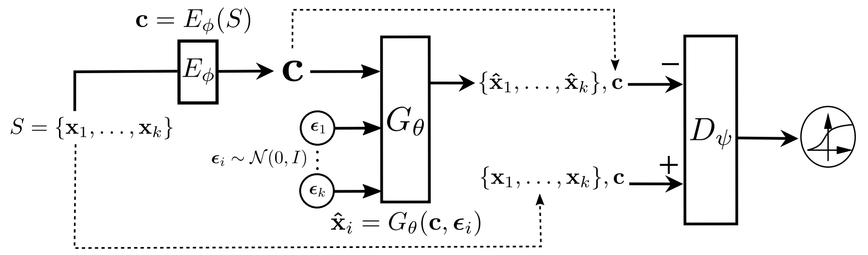
2 Related Work

Existing few-shot image generation methods can be broadly classified into two approaches: (a) fine-tuning the weights of pre-trained GANs on an unseen support set and (b) learning representations of support sets.

Fine-tuning the weights of pre-trained GANs One of the recent approaches of few-shot image generation is to fine-tune the weights of pre-trained GANs on novel datasets with few images [21, 28, 15, 19, 22]. These approaches work by preserving the diversity from the source or training domain while modifying the network such that target domain images can be produced. However, in these methods, fine-tuning the generative model often requires tedious manual designs [21, 28], and can also incur a high computational cost even when there is one example in the support set [15].

Learning representations of support sets Another approach for few-shot image generation is to learn a representation for the support sets. For example, [5] proposed the neural statistician, which aims at learning representations of datasets using variational inference. Another related work is [25], which proposed a sequential generative model that demonstrated the ability to synthesize new samples when given an example of a novel concept. Data-augmentation GAN (DAGAN) [1] is another few-shot generative model that utilizes the adversarial objective, designed specifically for using the generated samples for data augmentation. [24] proposed a conditional autoregressive generative model that can condition on the novel support set and generate new images.

These two approaches have wildly different trade-offs and applications. Since the first approach has a training phase on the new dataset, it can better adapt to datasets with different styles than the training set. However, it requires an expert ML practitioner to fine-tune the weights of the GAN on a new domain, has a higher computational cost and can only generalize to one new domain at a time. The second approach, which includes ISSA, can infer the representation of the new support set with only a feedforward pass, requires significantly less computational cost, can simultaneously generate images from different novel datasets, and learn representations that can be used in downstream tasks.


$\mathbf{c} = E_\phi(S)$
E_ϕ
$\mathbf{c}$
G_θ
$S = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$
ϵ_1
$\epsilon_i \sim \mathcal{N}(0, I)$
ϵ_k
$\hat{\mathbf{x}}_i = G_\theta(\mathbf{c}, \epsilon_i)$
$\{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_k\}, \mathbf{c}$
$\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}, \mathbf{c}$
$-$
$+$
D_ψ
Figure 2: Architecture of ISSA

Implicit Autoencoders [17] proposed the "implicit autoencoder" (IAE) which utilizes implicit distributions to learn expressive posterior and conditional likelihood distributions. The encoder in IAE extracts a context code from each input image, and the generator conditions on the context code and uses the noise vector to stochastically reconstruct the input. ISSA is inspired by IAE, but considers the problem of learning a generative model for datasets rather than individual data points. In particular, ISSA reduces to IAE when the support set size is one. However, as the size of support sets increases, ISSA's performance also improves. This is because learning from support sets enables ISSA to better disentangle the shared information of the support sets, from the information across the support sets.

3 Implicit Support Set Autoencoders

Problem statement Suppose we have a set of (not necessarily distinct) distributions $p_i(\mathbf{x})$ for $i \in \{1, \ldots, N\}$, and for each distribution we have a support set of data $S_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ consisting of a number of i.i.d. samples from $p_i(\mathbf{x})$. In the k-shot image generation task, we are given a collection of support sets $S_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ of size k, for $i \in \{1, \ldots, N\}$, and are asked to learn the distributions $\hat{p}_i(\mathbf{x})$ that closely approximate the underlying data distribution $p_i(\mathbf{x})$, for each i. At the test time, we are given a new support set $S_{\text{TEST}} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ whose elements are i.i.d. samples from a novel distribution $p_{\text{TEST}}(\mathbf{x})$, and the goal is to find a distribution $\hat{p}_{\text{TEST}}(\mathbf{x})$ that closely approximate $p_{\text{TEST}}(\mathbf{x})$.

Implicit support set generation In this work, we assume $\hat{p}_1(\mathbf{x}), \cdots, \hat{p}_N(\mathbf{x})$ are conditional implicit distributions parametrized by a single deep neural network g_θ. More specifically, each distribution $\hat{p}_i(\mathbf{x})$ is defined by a low dimensional representation $\mathbf{c}_i \in \mathbb{R}^l$ where we have:

$$\mathbf{x} = g_\theta(\boldsymbol{\epsilon}, \mathbf{c}_i) \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I). \tag{1}$$

For a given implicit distribution $\hat{p}_i(\mathbf{x})$, the distribution of $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ is the following factorial distribution:

$$\hat{p}_i(S) = \hat{p}_i(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \prod_{j=1}^{k} \hat{p}_i(\mathbf{x}_j), \qquad \text{or} \qquad \mathbf{x}_j = g_\theta(\boldsymbol{\epsilon}_j, \mathbf{c}_i) \quad \boldsymbol{\epsilon}_j \sim \mathcal{N}(0, I). \tag{2}$$

In other words, each $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ is generated by conditioning the implicit distribution on the representation $\mathbf{c}_i$, and passing a collection of k independent Gaussian noise $\mathcal{E} = \{\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_k\}$ through g_θ. For convenience, we use the notation $S = G_\theta(\mathcal{E}, \mathbf{c})$ to refer to this generative process for the support sets.

The goal of training is to learn the parameters of the neural network g_θ, as well as a representation $\mathbf{c}_i$ for each individual $\hat{p}_i(\mathbf{x})$, where $i \in \{1, \ldots, N\}$, such that $\hat{p}_i(\mathbf{x})$ closely approximate $p_i(\mathbf{x})$. In the generative process of $S_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ (Eq. 2), $\mathbf{c}_i$ is the shared representation, and thus it learns the common information across $\mathbf{x}_j$ in the support set, while each $\boldsymbol{\epsilon}_j$ learns to generate the information specific to each example $\mathbf{x}_j$.

Implicit support set autoencoders We now describe our proposed Implicit Support Set Autoencoders (ISSA) for k-shot image generation task. The architecture of ISSA can be seen in Fig. 2. Suppose we have a support set $S_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\} \sim p_i(S)$, whose elements come from the distribution $p_i(\mathbf{x})$. The first component of ISSA is a recognition path, which consists of an encoder neural network E_ϕ that takes the support set S_i as the input and predicts the context representation vector $\mathbf{c}_i = E_\phi(S_i)$ capturing the common information in S_i. The next component is the generative path that takes the predicted context vector $\mathbf{c}_i$ as the input and uses a collection of Gaussian noise $\mathcal{E} = \{\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_k\}$ to generate a support set $\hat{S}_i = G_\theta(\mathcal{E}, \mathbf{c}_i)$ according to Eq. 2. The goal of training is to learn E_ϕ and G_θ, such that conditioned on $\mathbf{c}_i$, the distribution of $\hat{S}_i \sim \hat{p}_i(S)$ is the same as the distribution of $S_i \sim p_i(S)$: $\hat{p}_i(S) = p_i(S)$. In order to ensure this, we use a discriminator network D_ψ that takes in $(S_i, \mathbf{c}_i)$ as the positive example and $(\hat{S}_i, \mathbf{c}_i)$ as the negative example and tries to discriminate them. This discriminator can then provide a gradient to train both the recognition path and the generative path. More specifically, the parameters of the ISSA are learnt by following a two-player minmax game with the value function:

$$\begin{aligned}\min_{\phi,\theta}\max_{\psi}\mathcal{L}(\phi,\theta,\psi) &= \mathbb{E}_{S\sim p(S)}\Big[\log D_\psi(S,\mathbf{c})\Big] + \mathbb{E}_{p(S)\hat{p}(\hat{S}|S)}\Big[\log\Big(1 - D_\psi(\hat{S},\mathbf{c})\Big)\Big] \\ &= \mathbb{E}_{S\sim p(S)}\Big[\log D_\psi\big(S, E_\phi(S)\big)\Big] + \mathbb{E}_{p(S)p(\mathcal{E})}\Big[\log\Big(1 - D_\psi\big(G_\theta(E_\phi(S),\mathcal{E}), E_\phi(S)\big)\Big)\Big].\end{aligned} \tag{3}$$

At the test time, we are given a new $S_{\text{TEST}} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ from the underlying distribution $p_{\text{TEST}}(\mathbf{x})$. We first pass S_{TEST} through the encoder network to predict the representation $\mathbf{c}_{\text{TEST}}$. We then condition on $\mathbf{c}_{\text{TEST}}$ and sample from the implicit distribution defined by $\hat{S} = G_\theta(\mathcal{E}, \mathbf{c}_{\text{TEST}})$ to generate new data. Implementation details of ISSA can be found in Appendix A.1.

Implicit autoencoding vs. standard autoencoding We now contrast the implicit autoencoding objective with the standard autoencoding objective. Suppose we have a standard autoencoder of support sets, which takes the support set S as the input, encode it using E_ϕ to obtain $\mathbf{c} = E_\phi(S)$, and then decode $\mathbf{c}$ using D_θ to obtain $\hat{S} = D_\theta(\mathbf{c})$. Further, suppose the objective of this autoencoder is the average reconstruction cost of each element: $\mathcal{L}(S, \hat{S}) = \frac{1}{k}\sum_{j=1}^{k} \|\hat{\mathbf{x}}_j - \mathbf{x}_j\|_2^2$. In this autoencoder, $\mathbf{c}$ must learn all the information in the set S, so that it can reconstruct every $\mathbf{x}_j$ perfectly. However, in ISSA, we perform a stochastic reconstruction (rather than deterministic reconstruction) of the support set, which requires $\hat{p}_i(S) = p_i(S)$. This enables the encoder E_ϕ to only learn the common information about the underlying distribution, rather than example-specific information. For example, if we have two support sets S_1, S_2, that come from the same distribution, the representations $\mathbf{c}_1 = E_\phi(S_1)$ and $\mathbf{c}_2 = E_\phi(S_1)$ in the standard autoencoder will be different. However, ISSA encourages $\mathbf{c}_1$ and $\mathbf{c}_2$ to have the same representation, identifying the underlying distribution. Thus, intuitively, the $\mathbf{c}$ in ISSA learns the relationship across distributions by capturing the *inter-distribution* information, while each conditional implicit distribution $\hat{p}_i(\mathbf{x})$ learns the *intra-distribution* relationship in an unsupervised fashion.

4 Experiments

We demonstrate ISSA's ability to generate diverse and high quality samples for novel classes on the CelebA [16] and the Omniglot [14] datasets. ISSA is only trained on the training domain and then evaluated on a disjoint set of classes in the test domain. We report the mean and standard deviation from three runs with different random seeds. Throughout this section, we use the term k-shot ISSA model to refer to the setting where the sizes of both training and test support sets are k. If the test support set size is different from the training support set size, we explicitly state that in parentheses.

Datasets The Omniglot dataset [14] contains 1623 different handwritten characters from 50 different alphabets. Each of these was hand drawn by 20 different people. It is a widely adopted benchmark for few-shot learning [5, 27, 1]. We use the same domain split as [1] with 1200 training classes and 211 test classes. CelebFaces Attributes Dataset (CelebA) is a large-scale human faces dataset with more than 200k celebrity images from 10,177 unique identities [16]. We use cropped images resized to 64×64 with the default training split [16] of 8000 train identities and 1000 test identities. We use Instance Selection [4] to retain 80% images in the most dense regions in the training domain.

Baselines For comparison, we consider two competitive few-shot image generation methods as baselines. First, the implicit autoencoder (IAE) [17] is capable of stochastic reconstruction of the input, thus able to generate varied outputs based on individual input samples. We use our own implementation of IAE with the same architecture as ISSA for fair comparison. Second, Data Augmentation GAN (DAGAN) [1] is a GAN based method to generate augmented data for few-shot

CelebA

Metrics	FID (↓)	Top 1 accuracy (↑)	Top 5 accuracy (↑)	Top 10 accuracy (↑)
IAE [17]	31.36 ± 0.08	0.0282 ± 0.0011	0.1445 ± 0.0645	0.2263 ± 0.0835
2-shot ISSA	28.97 ± 0.19	0.0177 ± 0.0004	0.0697 ± 0.0017	0.1205 ± 0.0026
10-shot ISSA (2-shot test)	19.30 ± 0.10	0.0542 ± 0.0014	0.1706 ± 0.0016	0.2663 ± 0.0012
5-shot ISSA	20.12 ± 0.27	0.0519 ± 0.0001	0.1742 ± 0.0013	0.2752 ± 0.0016
10-shot ISSA	$\mathbf{17.76 \pm 0.19}$	$\mathbf{0.0805 \pm 0.0013}$	$\mathbf{0.2349 \pm 0.0005}$	$\mathbf{0.3444 \pm 0.0030}$

Table 1: FID and identity accuracy result on CelebA.

Omniglot

Metrics	FID (↓)	Top 1 accuracy (↑)	Top 5 accuracy (↑)	Top 10 accuracy (↑)
DAGAN [1]	215.93 ± 6.59	$\mathbf{0.3853 \pm 0.0165}$	$\mathbf{0.6379 \pm 0.0262}$	0.7350 ± 0.0216
IAE [17]	161.28 ± 0.45	0.2709 ± 0.0021	0.5455 ± 0.0011	0.6737 ± 0.0007
2-shot ISSA	118.63 ± 0.27	0.2274 ± 0.0002	0.4957 ± 0.0003	0.6321 ± 0.0004
10-shot ISSA (2-shot test)	112.96 ± 0.49	0.2778 ± 0.0006	0.5537 ± 0.0002	0.6778 ± 0.0005
5-shot ISSA	113.18 ± 0.25	0.3239 ± 0.0007	0.6186 ± 0.0011	0.7432 ± 0.0007
10-shot ISSA	$\mathbf{109.07 \pm 0.23}$	0.3420 ± 0.0010	$\mathbf{0.6327 \pm 0.0010}$	$\mathbf{0.7488 \pm 0.0007}$

Table 2: FID and character accuracy result on Omniglot.

classification. We use the publicly available official implementation which only includes Omniglot experiments. Thus, DAGAN is not included in the CelebA comparison. Note that both IAE and DAGAN are one-shot generative models where each sample from the support set is treated individually rather than jointly as in ISSA. In addition, note that ISSA with one shot (i.e., $k = 1$) reduces to IAE.

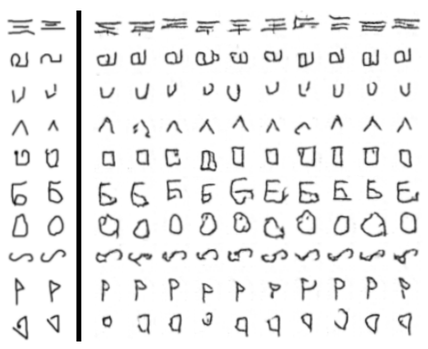
4.1 Quantitative Evaluation

Evaluating generative models remains an open problem. A successful k-shot image generation method should generate realistic and diverse samples that accurately describe the target distribution. We discuss two metrics to evaluate the diversity and accuracy of the generated samples respectively.

Diversity evaluation We use the widely adopted "Fréchet Inception Distance" (FID) [7] to measure the quality and diversity of the generated samples. As the generated samples are conditioned on real images from the support set, we only use real images that are not in the support set for FID evaluation. Therefore, for a given test class, we set aside half of the images to randomly sample support sets and the other half as the real-world image for FID computation. For Omniglot, each class has 20 images thus we use 10 of those to sample the support sets and the remaining 10 images for FID evaluation (i.e., with 210 test classes, 2110 images are used to compute the FID). For CelebA, since each identity has varied number of images, we select classes with more than 20 images and again use the first 10 as support set samples and another 10 for FID. We use 10 generated images per class based on 10 real samples to be compared against the 10 held-out real images of that class. To ensure that the evaluation is fair for any k-shot ISSA, each real image is used exactly once to form the support set. For example, for 10-shot ISSA, a support set of 10 real images is used to obtain one context vector $\mathbf{c}$ and used to generate 10 images. In contrast, for a one shot model such as IAE, 10 support set of size one are formed and each generates one image.

Accuracy evaluation We also examine how closely the generated samples resemble the underlying distribution of the support set (using an *identity accuracy* metric). To achieve this, we train a classifier on the test classes and check if the generated samples can fool the classifier. In practice, we randomly sample 10 support sets from each class and generate 10 images from each support set. These 100 images are then checked by the classifier to see if they belong to the same class as the support set. This process is applied to all classes in the Omniglot test domain and the 200 most frequent classes from the CelebA test domain.

The diversity and accuracy evaluation are complementary metrics in evaluating k-shot image generation methods. For instance, a standard autoencoder can achieve close to perfect accuracy score as it reconstructs the support set, but obtains a poor performance on the diversity metric. More details regarding the backbone classifier for both evaluation metrics can be found in Appendix A.2.


(a) Test support set size of $k = 2$.

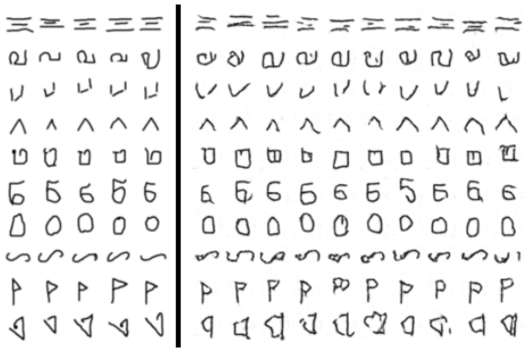

(b) Test support set size of $k = 5$.

Figure 3: ISSA generates high quality and diverse images for unseen classes with as few as (a) 2 images and (b) 5 images in Omniglot. As the size of the support set increases, ISSA can infer a better representation of the underlying distribution. The support set images are placed at the left side of the vertical bar and the generated images are on the right.

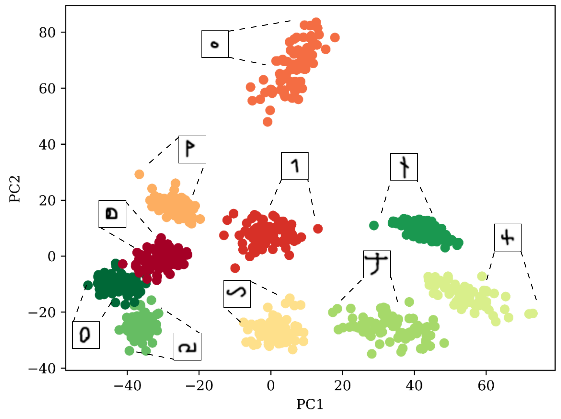


Figure 4: 2D PCA visualization of the context vector **c** of the 10-shot ISSA for 10 test classes of the Omniglot dataset. Each dot is the context vector of a support set of size 10 and is colored according to its class.

Results We train the k-shot ISSA with varying number of shots $k \in \{2, 5, 10\}$ on the CelebA and Omniglot datasets, and report the accuracy and diversity of generated images in Table 1 and Table 2 (note that ISSA with $k = 1$ is equivalent to IAE). The top results within the error margin are highlighted.

We observe that as we increase k, both the diversity and accuracy of k-shot generated samples improve consistently in both CelebA and Omniglot datasets, with the 10-shot ISSA achieving the best FID and identity classification results. This is due to the fact that by increasing k, ISSA can better capture the underlying distribution by learning the common information within the sets. This is in contrast to IAE and DAGAN, which attempt to learn the underlying distributions from individual examples rather than sets. On the CelebA dataset, ISSA achieves significant gains in both FID and accuracy, compared to IAE. On the Omniglot dataset, the ISSA also achieves significant improvements in terms of FID, outperforming IAE and DAGAN. The 10-shot ISSA outperforms IAE by 52.21 and DAGAN by 106.86 in term of the FID score. For identity accuracy, 10-shot ISSA outperforms IAE and remains competitive with DAGAN.

We also examine our 10-shot ISSA, in the setting where the model is tested on sets of size $k = 2$: 10-shot ISSA (2-shot test) in Table 1 and Table 2. In order to do so, during the test time, we simply repeat examples from the support set to form a set of size 10. As expected, we observe that this model performs better than ISSA trained in the 2-shot setting. This is because training on larger support

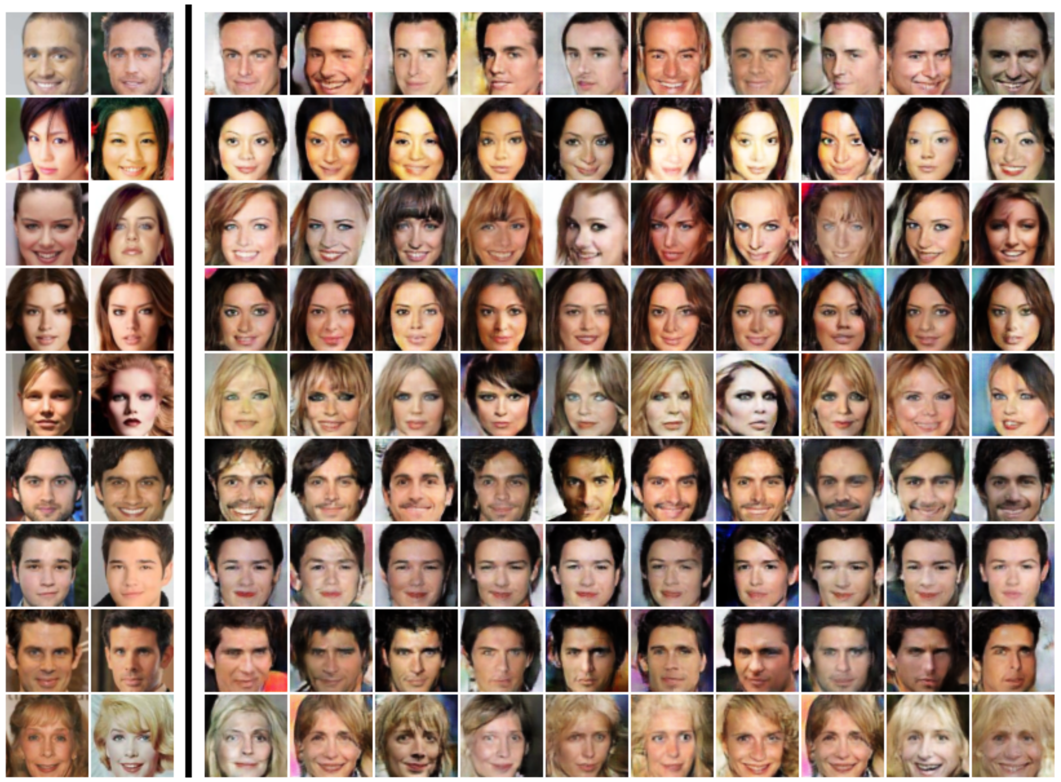


Figure 5: ISSA generates high quality and diverse images for unseen classes with as few as 2 images in CelebA. The support set images are placed at the left side of the vertical bar and the generated images are on the right.

set has enabled ISSA to learn better representations for the underlying distributions. However, it performs worse than the 10-shot ISSA, as at the test time it has not seen as many diverse images to accurately infer the underlying distribution. Therefore, our overall recommendation is to train ISSA with large support set sizes (i.e. size 10) and use it for support set of equal or smaller sizes at test time.

4.2 Qualitative Evaluation

We now visualize the generated samples from ISSA on both the CelebA and Omniglot datasets. The vertical bars separate the support set images from the samples generated by ISSA. Figure 3 shows the samples generated by 10-shot ISSA from test support sets with only 2 examples (a) or 5 examples (b) from the Omniglot dataset. We see that ISSA successfully learns the global structure of the characters and generate new diverse images from the same class. As the support set size increase, ISSA can better capture the underlying distribution of the support set, thus generating samples that are more representative of the dataset. The generated samples are of high visual quality with diverse variations.

Figure 5 shows CelebA generations from ISSA with only 2 support examples. The context vector $\mathbf{c}$ obtained from the encoder attempts to infer the underlying distribution of the support set and the noise vector controls the variations in the output image. Our best CelebA model can generate images for new identities with the accuracy of 8% top-1 and 34% top-10 across 1000 test identities, as measured by a strong identity classifier (Table 1). Compared to IAE and DAGAN, this is a significant improvement, however, this task still remains a challenging problem. We can see from Figure 5 that the context vector of ISSA mostly captures high-level features relevant to the identity of the face (e.g., gender, ethnicity, shape of eye brows), and that the noise vector ϵ captures low-level variations independent of the identity (e.g., head pose, lighting condition).

4.3 Context Space

In order to assess how effectively the ISSA encoder infers the underlying distribution of a given support set, we visualize the context vectors of 80 support sets of size 10 randomly drawn from 10 test classes of the Omniglot dataset in Figure 4. Each dot is colored based on its class label. We observe that support sets from the same underlying distribution (i.e., same character in Omniglot) are mapped to the same region in the PCA projection space, forming distinct clusters. Interestingly, characters with similar structures also lie in closer proximity in the context space. This suggests that ISSA successfully learns to capture the inter-class information from the training domain in a way that can be transferred to new classes.

5 Conclusion

In this work, we proposed the "Implicit Support Set Autoencoder" (ISSA) for few-shot image generation tasks. We showed that ISSA can learn useful representations for the underlying distributions of support sets with an adversarial objective. This context representation can then be leveraged at the test time to infer the underlying distribution of a novel dataset, and generate new samples from it. We evaluate the performance of ISSA on CelebA and Omniglot datasets using both accuracy and diversity metrics. On the Omniglot dataset, we showed that ISSA can learn the global structure of the characters and generate new characters with diverse variations while preserving the global structure. On the CelebA dataset, we showed that ISSA can perform better than recent methods of IAE and DAGAN in inferring novel identities from few examples, and generating diverse images from them.

Acknowledgments and Disclosure of Funding

Alireza Makhzani and Guillaume Rabusseau acknowledge support from the Canada CIFAR AI Chairs program.

References

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[3] Jiankang Deng, Jia Guo, Zhou Yuxiang, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-stage dense face localisation in the wild. In *arxiv*, 2019.

[4] Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *Advances in Neural Information Processing Systems*, 33:13285–13296, 2020.

[5] Harrison Edwards and Amos Storkey. Towards a neural statistician. In *International Conference on Learning Representations*, 2017.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[8] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[9] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[10] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[11] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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

[14] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[15] Yijun Li, Richard Zhang, Jingwan Cynthia Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. In *Advances in Neural Information Processing Systems*, 2020.

[16] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[17] Alireza Makhzani. Implicit autoencoders. *arXiv preprint arXiv:1805.09804*, 2018.

[18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[19] Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Freeze discriminator: A simple baseline for fine-tuning gans. *arXiv preprint arXiv:2002.10964*, 2020.

[20] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[21] Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2750–2758, 2019.

[22] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10743–10752, 2021.

[23] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[24] Scott Reed, Yutian Chen, Thomas Paine, Aäron van den Oord, SM Ali Eslami, Danilo Rezende, Oriol Vinyals, and Nando de Freitas. Few-shot autoregressive density estimation: Towards learning to learn distributions. In *International Conference on Learning Representations*, 2018.

[25] Danilo Rezende, Ivo Danihelka, Karol Gregor, Daan Wierstra, et al. One-shot generalization in deep generative models. In *International Conference on Machine Learning*, pages 1521–1529. PMLR, 2016.

[26] Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. `https://github.com/mseitzer/pytorch-fid`, August 2020. Version 0.1.1.

[27] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3637–3645, 2016.

[28] Yaxing Wang, Abel Gonzalez-Garcia, David Berga, Luis Herranz, Fahad Shahbaz Khan, and Joost van de Weijer. Minegan: effective knowledge transfer from gans to target domains with few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9332–9341, 2020.

A Hyperparameters

A.1 ISSA Architecture

In this section, we provide information regarding the architectural implementation of ISSA. In the implementation of ISSA, we use DCGAN [23] inspired architectures for the encoder, the generator and the discriminator. We use spectral normalization [18] to stabilize training for the generator and the discriminator. The exact architectures for the encoder E, the generator G and the discriminator D can be found in Table 3, 4 (for Omniglot) and Table 5, 6 (for CelebA). In these tables, "Conv" stands for 2D convolution layer, "Pad" stands for amount of padding in the convolution layer, "BN" stands for a batch normalization layer [10], "SNConv" stands for 2D convolution layer with spectral normalization, "SNConvTr" stands for 2D transposed convolution with spectral normalization.

The encoder is implemented as a CNN and outputs the context vector $\mathbf{c}$. To pass the entire support set S_i to the encoder, the images within the set are stacked as additional channels to the first convolutional layer. For example, in 5-shot ISSA on the CelebA dataset, the input to the encoder is a set of five 64×64 color images. They are then stacked to form a tensor of size $\mathbb{R}^{15\times 64\times 64}$ before being fed to the first convolutional layer. The output representation from the last convolutional layer of the CNN is fed into two fully connected layers which outputs the context vector $\mathbf{c}$. The discriminator D is also a CNN which takes a set of images as input. Similar to the encoder, the images from a set are stacked as additional input channels. To condition on the context $\mathbf{c}$, we embed $\mathbf{c}$ with a fully connected hidden layer and then concatenate the embedding as an additional channel to the stacked images. Thus, using the above example, the input to the first convolutional layer in the discriminator D would be of size $\mathbb{R}^{16\times 64\times 64}$. Lastly, the generator G takes both the context $\mathbf{c}$ and the noise vector $\mathbf{z}$ as input. As both $\mathbf{z}$ and $\mathbf{c}$ are vectors, we simply concatenate them to form the input to the generator.

Encoder	Generator
$S \in \mathbb{R}^{32\times 32\times k}$	$\mathbf{c} \in \mathbb{R}^{1000}$ and $n \in \mathbb{R}^{1000}$
4×4 Conv.100 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 400 ReLU. Stride 1. BN
4×4 Conv.200 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 200 ReLU. Stride 2. Pad 1. BN
4×4 Conv.400 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 100 ReLU. Stride 2. Pad 1. BN
FC. 2000 Linear ReLU BN	4×4 SNConvTr.1 Tanh. Stride 2. Pad 1.
FC. 1000 Linear ReLU BN	

Table 3: Omniglot ISSA encoder and generator architectures

Discriminator
(S, c) or $(\hat{S}, c)$ where $S \in \mathbb{R}^{32\times 32\times k}$ and $\mathbf{c} \in \mathbb{R}^{1000}$
4×4 SNConv.100 LeakyReLU. Stride 2. Pad 1. BN
4×4 SNConv.200 LeakyReLU. Stride 2. Pad 1. BN
4×4 SNConv.400 LeakyReLU. Stride 2. Pad 1. BN
Sum Pooling Sigmoid.

Table 4: Omniglot ISSA discriminator architecture

Here, We describe the training hyperparameters. We use "c scale" to refer to the scaling factor for the context vector $\mathbf{c}$ and "n scale" to refer to the scaling factor for the noise vector $\boldsymbol{\epsilon}$ (is set to be 1 if not specified). For ISSA training, we use Adam optimizer. First, we describe the hyperparameters for the Omniglot dataset. For 2-shot ISSA, the learning rate for E, G and D is set to 2×10^{-4} and c scale is 2×10^{-1}. For 5-shot ISSA, the learning rate for E, G is 2×10^{-4} and for D is 10^{-4} while c scale is 2×10^{-1}. For 10-shot ISSA, the learning rate for E, G and D is set to 2×10^{-4} and c scale is 10^{-1}. For IAE, the learning rate for E, G and D is set to be 2×10^{-4} and c scale is 10^{-1}. On the CelebA dataset, for 2-shot ISSA, the learning rate for E, G is 3×10^{-4} and for D is 2×10^{-4} and c scale is 2×10^{-4}. And 5-shot ISSA has the same parameter as 2-shot ISSA. For 10-shot ISSA the learning rate for E, G is 3×10^{-4} and for D is 2×10^{-4} and c scale is 10^{-1}. Now we describe the parameters for the baselines in 4. For IAE, the learning rate for E, G is 3×10^{-4} and for D is

Encoder	Generator
$S \in \mathbb{R}^{64\times 64\times 3\times k)}$	$\mathbf{c} \in \mathbb{R}^{100}$ and $n \in \mathbb{R}^{100}$
4×4 Conv.64 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 512 ReLU. Stride 1. BN
4×4 Conv.128 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 256 ReLU. Stride 2. Pad 1. BN
4×4 Conv.256 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 128 ReLU. Stride 2. Pad 1. BN
4×4 Conv.512 LeakyReLU. Stride 2. Pad 1. BN	4×4 SNConvTr. 64 ReLU. Stride 2. Pad 1. BN
FC. 2000 Linear ReLU BN	4×4 SNConvTr.3 Tanh. Stride 2. Pad 1.
FC. 100 Linear ReLU BN	

Table 5: CelebA ISSA encoder and generator architectures

Discriminator
(S, c) or $(\hat{S}, c)$ where $S \in \mathbb{R}^{64\times 64\times 3\times k}$ and $\mathbf{c} \in \mathbb{R}^{100}$
4×4 SNConv.64 LeakyReLU. Stride 2. Pad 1. BN
4×4 SNConv.128 LeakyReLU. Stride 2. Pad 1. BN
4×4 SNConv.256 LeakyReLU. Stride 2. Pad 1. BN
4×4 SNConv.512 LeakyReLU. Stride 2. Pad 1. BN
Sum Pooling Sigmoid.

Table 6: CelebA ISSA discriminator architecture

2×10^{-4} while c scale is 1 and n scale is 2×10^{-1}. For DAGAN, we use the default parameter from the official implementation.

A.2 Evaluation Details

In this section, we describe the backbone classifiers used for both FID and identity accuracy evaluation in Section 4. During training, we use the FID for the training domain as indicator for when to stop training ISSA.

Omniglot Both evaluation backbone classifiers take 32×32 images as inputs and when applicable, we resize images into 32×32 (such as the output from DAGAN). We use a CNN trained with all 1623 Omniglot characters as backbone for FID evaluation. The detailed architecture can be found in Table 7. The classifier is trained with 18 training examples per class, using early stopping with 2 validation examples per class. The learning rate is set to be 1×10^{-3} with Adam optimizer [12] ($\beta_1 = 0.9$ and $\beta_2 = 0.999$).

For identity classification, we use a DenseNet [8] classifier trained on the test domain for 211-way classification on the test classes. The classifier is trained using 18 images per class and achieves 99% validation accuracy on 1 held-out example and 96.17% test accuracy on 1 held-out example. The DenseNet classifier has 4 Dense Blocks and 4 Transition Layers with a growth rate of 64. Each Dense Block has 3 convolution layers within it. The first convolution layer has 64 filters and we apply a dropout rate of 0.5. The classifier is trained with learning rate of 0.001 with Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ and we use early stopping with validation examples.

Omniglot FID backbone classifier
$x \in \mathbb{R}^{32\times 32\times 1}$
4×4 Conv.64 LeakyReLU. Stride 2. Pad 1. BN
4×4 Conv.128 LeakyReLU. Stride 2. Pad 1. BN
4×4 Conv.256 LeakyReLU. Stride 2. Pad 1. BN
FC. 2000 Linear ReLU BN
FC. 1623 Linear ReLU BN

Table 7: Omniglot FID backbone

CelebA The FID backbone is computed with a pretrained InceptionV3 classifier [26] [1]. The backbone for identity classification is a pretained InsightFace classifier [3] [2]. It achieves 96% test accuracy with 29 training examples per class from the CelebA test domain.

B Computational Resources

To run all experiments in Section 4, we use either one Nvidia T4 or P100 or RTX6000 GPU with 64G memory and 4 CPU cores. Here, we will describe the estimated compute time for ISSA. At the test time, inferring the underlying distribution from a support set and producing novel samples is a simple forward pass on ISSA thus being near instant (when running with GPU). Training ISSA on Omniglot takes less than 6 hours, and on CelebA it would take less than a day.

[1] `https://github.com/mseitzer/pytorch-fid`

[2] `https://github.com/deepinsight/insightface`