Supplementary: Towards Zero-Shot Learning with Fewer Seen Class Examples

Vinay Kumar Verma[1*] Ashish Mishra[2*] Anubha Pandey[2$]
Hema A. Murthy[2] Piyush Rai[1]
[1]Department of CSE, IIT Kanpur; [2]Department of CSE, IIT Madras,
{vkverma,piyush}@cse.iitk.ac.in; {mishra,hema}@cse.iitm.ac.in; anubhap93@gmail.com

Algorithm 1 Meta-VGAN for ZSL

Require: $p(\mathcal{T})$:distribution over tasks
Require: η_1, η_2: step-size hyperparameters
1: Randomly initialize $\theta_e, \theta_g, \theta_d$
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$; where, $\mathcal{T}_i = \{\mathcal{T}_i^{tr}, \mathcal{T}_i^{v}\}$ such that $\mathcal{T}_i^{tr} \cap \mathcal{T}_i^{v} = \phi$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_{\theta_{eg}} l^{VG}_{\mathcal{T}_i^{tr}}(\theta_{eg})$
6: Evaluate $\nabla_{\theta_d} l^{D}_{\mathcal{T}_i^{tr}}(\theta_d)$
7: Compute adapted parameters: $\theta^{'}_{eg} = \theta_{ed} - \eta_1 \nabla_{\theta_{eg}} l^{VG}_{\mathcal{T}_i^{tr}}(\theta_{eg})$
8: Compute adapted parameters: $\theta^{'}_{d} = \theta_{d} + \eta_2 \nabla_{\theta_{d}} l^{D}_{\mathcal{T}_i^{tr}}(\theta_{d})$
9: Update $\theta_{eg} \leftarrow \theta_{eg} - \eta_1 \nabla_{\theta_{eg}} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} l^{VG}_{\mathcal{T}_i^{v}}(\theta^{'}_{eg})$
10: Update $\theta_{d} \leftarrow \theta_{d} + \eta_2 \nabla_{\theta_{d}} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} l^{D}_{\mathcal{T}_i^{v}}(\theta^{'}_{d})$

1. Datasets Descriptions

Dataset	Attribute/Dim	#Image	Seen/Unseen Class
AwA2[3]	A/85	37322	40/10
CUB[2]	CR/1024	11788	150/50
SUN[4]	A/102	14340	645/72
aPY[1]	A/64	15339	20/12

Table 1: The benchmark datasets used in our experiments, and their statistics.

To evaluate our proposed model in comparison with several state-of-the-art ZSL and generalized ZSL methods, we applied our approach to the following benchmark ZSL datasets: SUN[4], CUB[2], AwA2[3], and aPY [1]. Table 1 shows the summary of the datasets used and their statistics.

SUN Scene Recognition: SUN is a fine-grained dataset with 717 scene categories and 14,340 images. We use the widely used split of the dataset for the ZSL setting, 645 seen classes, and 72 unseen classes. The dataset has image-level attributes. For training, we use class-level attributes obtained by combining the attributes of all the images in a class.

Animals with Attributes: AwA2 is a coarse-grained dataset with 50 classes and 37,322 images. We follow a standard zero-shot split of 40 seen (train) classes and ten unseen (test) classes. The dataset has 85-dimensional human-annotated class-attributes.

Caltech UCSD Birds 200: CUB is a fine-grained dataset with 11,788 images from 200 different types of birds, annotated with 312 attributes. We use a zero-shot split of 150 unseen and 50 seen classes. The dataset has image-level attributes like the SUN dataset. We average these image-level attributes of all the classes to obtain class attributes for training.

Attribute Pascal and Yahoo (aPY): aPY is a coarse-grained dataset with 64 attributes. The dataset has 32 classes. For Zero-Shot learning, we follow a split of 20 Pascal classes for training and 12 Yahoo classes for testing.

2. Implementation Details

Our proposed architecture Meta-VGAN has an encoder, decoder, generator, and discriminator modules, as shown in Figure-1 in the main paper. The decoder and generator modules share the same network parameters. Each of the modules has a series of FC layers followed by a ReLU and dropout layers. We concatenate image feature vector $\mathbf{x}$ extracted from ResNet-101 with class attributes vector $\mathbf{a}$ and feed to the Encoder module E. The encoder E has a series of three FC layers, and encodes the input to $\mathbf{d_z}$ (varies with datasets) dimensional latent space with mean μ and variance $\mathbf{\Sigma}$. Noise dimensions used for CUB, SUN, AwA2, and aPY datasets are 512, 20, 40, and 20, respectively. Next, we sample $\mathbf{d_z}$ dimensional noise vector from the latent space and feed it to the decoder module (or the generator module). The decoder (or generator) uses a series of 2 FC layers fol-

*Equal contribution. $ Currently author is affiliated with MasterCard

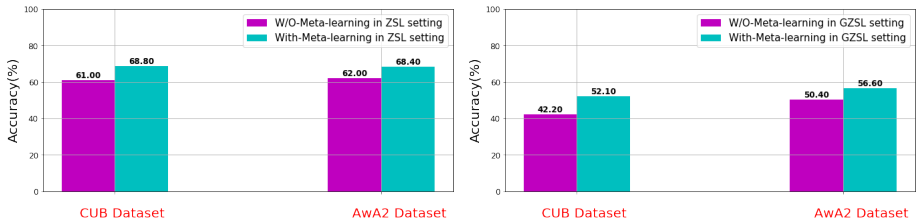

Figure 1: The left figure shows the mean per class accuracy with and without meta-learner in the ZSL setting. In the right figure the GZSL result are shown with and without meta-learning.

lowed by ReLU to generate a 2048 dimension feature vector $\hat{\mathbf{x}}$ similar to the input image feature vector $\mathbf{x}$. The generated image features $\hat{\mathbf{x}}$ are further passed through the discriminator module. The discriminator receives two types of inputs: the real image feature vector $\mathbf{x}$ that comes from the ground truth data of the training set and the synthesized image features $\hat{\mathbf{x}}$ generated by the generator module(or the decoder module). The discriminator has a series of 3 FC layers and tries to distinguish between the real image feature vector $\mathbf{x}$ and the generated image feature vector $\hat{\mathbf{x}}$. The discriminator outputs the probability of the image is real. The output value should be close to 0 for fake image features $\hat{\mathbf{x}}$, and it should be close to 1 for real image features $\mathbf{x}$.

For training, we associate each of the modules with a meta-learner agent in the Meta-VGAN model. We randomly sample 10 classes for training and ten classes for validation from the seen classes of the dataset such that they are mutually exclusive. We call this a *task*. We randomly sample 5 examples from each class of the train set and three examples from each class of the validation set. For each task, we iterate through each class of the train set multiple times and compute the adapted parameters of the network using Eq.7 and 8 in the main paper. Next, we pass the validation data through the network with initial parameters and with the computed parameters and compute the loss. We finally update the network on the validation loss, as shown in Eq.10 and 11 in the main paper. The learning rate and dropout rate used for all the datasets are 0.001 and 0.3, respectively. All the hyperparameters are selected using cross-validation.

The values of hyper-parameters η_1 and η_2, used for computation of updated parameters on training loss, are empirically chosen using a grid search in the range $[1e-1, 1e-8]$.

2.1. Comparison with and without meta-learning

Our model is a combination of CVAE and CGAN, which are integrated with a meta-learner. To illustrate the contribution of the meta-learner module, we perform an experiment when no meta-learning component is present. Figure 1 shows the comparison between with and without meta-learner in our model, for CUB and AWA2 datasets, in both standard ZSL and GZSL settings. We observe that the role of a meta-learner in our proposed model is very crucial, which helps to train our model very efficiently using a few examples per seen class. The meta-learner component boosts the model's absolute performance by 7.8% and 6.4% in standard ZSL, while by 9.9% 6.2% in GZSL for CUB and AWA2 datasets, respectively.

References

[1] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1785. IEEE, 2009.

[2] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[3] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[4] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492. IEEE, 2010.