Supplementary: Meta-Learned Attribute Self-Interaction Network for Continual and Generalized Zero-Shot Learning

Vinay Verma[2⋆], Nikhil Mehta[2⋆], Kevin J Liang[3], Aakansha Mishra[4], Lawrence Carin[5]
[2]Duke University, [3]FAIR, Meta, [4]IITG, [5]KAUST
{[1]vverma.vinayy,[2]nikhilmehta.dce}@gmail.com

1. Proofs

Lemma 1 (Polynomial Approximation) *Consider a model with L layers of self-interaction modules with parameters $\{\Phi_a^\ell,\, \Phi_s^\ell,\, \Phi_b^\ell\}_{\ell=1}^L$ and identity activation $g_a^\ell(x) = g_s^\ell(x) = g_b^\ell(x) = x$. Let input to the model be: $\mathbf{a} = [a_1, a_3, \dots, a_D]$. Then, the output of the model $\mathbf{a}_L$ approximates following class of polynomial functions:*

$$\left\{ P_\ell\left(\mathbf{a}\right) = \sum_\beta w_\beta\, a_1^{\beta_1}\, a_2^{\beta_2} \dots a_D^{\beta_D} \;\middle|\; 0 \le |\beta| \le 2^\ell \right\}, \tag{1}$$

where the sum is across multiple terms (monomials), $\beta = [\beta_1, \dots, \beta_D]$ is a vector containing the exponents of each attribute in a given term having degree $|\beta| = \sum_{i=1}^D \beta_i$, and w_β is the coefficient of the corresponding term that depends on the module parameters. Furthermore, the degree of the polynomial grows exponentially with the model depth.

Proof. For the polynomial approximation, the self-interaction module is defined as:

$$\mathbf{a}_{\ell+1} = \Phi_a^{\ell+1}\left(\mathbf{a}_\ell\right) * \Phi_s^{\ell+1}\left(\mathbf{a}_\ell\right) + \Phi_b^{\ell+1}\left(\mathbf{a}_\ell\right), \tag{2}$$

where $\mathbf{a}_0 = \mathbf{a} = [a_1, a_3, \dots, a_D]$ and $*$ is the element-wise multiplication operator. For the analysis, we consider $\{\Phi_a^\ell,\, \Phi_s^\ell,\, \Phi_b^\ell\}_{\ell=1}^L \in \mathbb{R}^{D\times D}$ as square matrices for all $\ell \in [1, \dots, L]$, however, the analysis can be easily extended to the case when these matrices are rectangular. To show that the output of the ℓ^{th} self-interaction module approximates the polynomial, we show that each coordinate of $\mathbf{a}_\ell$ belongs to the class of polynomials (1). In the following analysis, we denote the i^{th} vector coordinate as $\mathbf{a}_\ell[i]$. Similarly, the j^{th} column vector in the parameter matrix Φ^ℓ is denoted as $\Phi^\ell[:, j]$. Then, the proof of the lemma follows from induction:

Base: Consider the base case for $\mathbf{a}_0 = [a_1, a_3, \dots, a_D]$. Clearly, each coordinate $\mathbf{a}_0[i]$ belongs to the class of polynomials in (1). In particular, $\mathbf{a}_0[i] = a_i \in \{P_0\left(\mathbf{a}\right)\}$.

Induction step: Assume that when $\ell = k$, $\mathbf{a}_k[i] \in \{P_k(a)\}$ $\forall i \in \{1, \dots, D\}$. Then, for $\ell = k+1$, we have:

$$\mathbf{a}_{k+1} = \Phi_a^{k+1}(\mathbf{a}_k) * \Phi_s^{k+1}(\mathbf{a}_k) + \Phi_b^{k+1}(\mathbf{a}_k) \tag{3}$$

$$= \underbrace{\left(\sum_m \mathbf{a}_k[m]\, \Phi_a^{k+1}[:, m]\right)}_{r_a} * \underbrace{\left(\sum_n \mathbf{a}_k[n]\, \Phi_s^{k+1}[:, n]\right)}_{r_s} + \underbrace{\left(\sum_m \mathbf{a}_k[m]\, \Phi_b^{k+1}[:, m]\right)}_{r_b} \tag{4}$$

In (4), each coordinate of the vectors $\{r_a, r_s, r_b\} \in \mathbb{R}^D$ belongs to the class of polynomials $\{P_k(a)\}$ since $\mathbf{a}_k[i] \in \{P_k(a)\}$ $\forall i \in \{1, \dots, D\}$. Then, the i^{th} coordinate of $\mathbf{a}_{k+1}$ can be simplified as:

$$a_{k+1}[i] = r_a[i] * r_s[i] + r_b[i], \text{ where} \tag{5}$$

$$r_a[i] * r_s[i] = \sum_m \sum_n \mathbf{a}_k[m]\, \mathbf{a}_k[n]\, \Phi_a^{k+1}[i, m]\, \Phi_s^{k+1}[i, n] \tag{6}$$

$$\text{and } r_b[i] = \sum_m \mathbf{a}_k[m]\, \Phi_b^{k+1}[i, m] \tag{7}$$

Since (6) is the sum of the product of polynomials, it follows that the resulting $\mathbf{a}_{k+1}[i]$ is a polynomial. Moreover, the degree of $\mathbf{a}_{k+1}[i]$ satisfies the following:

$$\deg(\mathbf{a}_{k+1}[i]) \le \max_{m,n} \left[\deg(\mathbf{a}_k[m]) + \deg(\mathbf{a}_k[n])\right] \tag{8}$$

$$\le 2^k + 2^k = 2^{k+1} \tag{9}$$

Hence, $\mathbf{a}_{k+1}[i] \in \{P_{k+1}(a)\}$ $\forall i \in \{1, \dots, D\}$.

Lemma 2 (Maximize Entropy with IR) *Let $t_\xi(a|z) = \mathcal{N}(a; \mathcal{R}_\xi(z), I)$ be the probabilistic inverse map associated with the attribute encoder f_Φ, where $z = f_\Phi(a)$ denotes the attribute embedding. The mutual information between the attribute a and the attribute embedding z is defined as:*

$$I(a; z) = H\left(z; \Phi\right) \ge H\left(a\right) + \mathbb{E}_{a\sim p(a)}\left[\log t_\xi(a|f_\Phi(a))\right]. \tag{10}$$

Proof. Consider the attribute $a \in \mathbb{R}^D$ and the embedding vector $z \in \mathbb{R}^d$ to be random variables under $p_\Phi(a, z) =$

$p(a)\, p_\Phi(z|a)$ as the joint distribution. The mutual information between attribute and $I(a; z) = H(z; \Phi) - H(z|a) = H(a) - H(a|z; \Phi)$. As $f_\Phi : \mathbb{R}^D \rightarrow \mathbb{R}^d$ is a deterministic mapping, $p_\Phi(z|a)$ is a deterministic function of a, *i.e.* $p_\Phi(z|a) = \delta(z - f_\Phi(a))$. Hence, the conditional entropy $H(z|a) = 0$, and $H(z; \Phi) = H(a) - H(a|z; \Phi)$.

$$\begin{aligned} H(a|z;\Phi) &= -\,\mathbb{E}_{p_\Phi(a,z)} \log p_\Phi(a|z) \\ &= -\,\mathbb{E}_{p_\Phi(a,z)} \log t_\xi(a|z) - \mathbb{E}_{p_\Phi(a,c)} \log \frac{p_\Phi(a|z)}{t_\xi(a|z)} \\ &= -\,\mathbb{E}_{p_\Phi(a,z)} \log t_\psi(a|z) \\ &\quad -\,\mathbb{E}_{p(z)} \left[KL\left[p_\Phi(a|z), t_\xi(a|z)\right]\right] \\ &\leq -\,\mathbb{E}_{p_\Phi(a,z)} \log t_\xi(a|z) \end{aligned} \tag{11}$$

This inequality can be used to bound the entropy:

$$H(z;\Phi) = H(a) - H(a|z;\Phi) \tag{12}$$

$$\geq H(a) + \mathbb{E}_{p_\Phi(a,z)} \log t_\xi(a|z) \tag{13}$$

2. Datasets

We conduct experiments on five widely used datasets for zero-shot learning. CUB-200 [12] is a fine-grain dataset containing 200 classes of birds, and AWA1 [7] and AWA2 [13] are datasets containing 50 classes of animals, each represented by an 85-dimensional attribute. aPY [1] is a diverse dataset containing 32 classes, each associated with a 64-dimensional attribute. SUN [9] includes 717 classes, each with only 20 samples; fewer samples and a high number of classes make SUN especially challenging. In the SUN dataset, each class is represented by a 102-dimensional attribute vector. The train/test split details are given in Table 1 and the same split is used for the generalized zero-shot Learning (GZSL) setting.

The pre-processed dataset is provided by [13] and publicly available of the download[1]. The dataset use ResNet-101 architecture pretrained on the ImageNet [10] for the feature extraction of the visual domain. The features are directly extracted from the pretrained model without any finetuning. Also, the seen and unseen split proposed by [13] ensures that unseen classes are not present in the ImageNet dataset; otherwise, the zero-shot learning setting will be violated.

3. Training and Evaluation Protocols

In the training, first, we divide the training classes into train and validation sets as mentioned in Table-1. We tune the hyperparameter for the validation set that is discussed below. Once we have the optimal hyperparameter for the validation set we merge the train and validation set and retrain the model with the tuned hyperparameter and the model is evaluated for the test samples. The hyperparameters are tuned for the Generalized Zero-Shot Learning (GZSL) only and same parameters are used for all the other experiments like Zero-shot Learning, Fixed Continual GZSL, and Dynamic Continual GZSL.

We have three hyperparameters: λ, η, and ϵ. We search loss weight λ in the interval $[0.5, 10]$ with step size 0.5. Learning rate η is swept from 10^{-6} to 10^{-1} by a factor of 10, with learning rate decay with each epoch. We search Reptile learning rate ϵ between $[10^{-4}$, $10^{-1}]$. The final obtained hyperparameter are given in Section 3.1.1. Note our baseline results are reported from [2–4, 6]; we follow the same settings and split.

3.1. Generalized Zero-Shot Learning (GZSL)

The simplest case we consider is the generalized zero-shot learning (GZSL) setting [13]. In GZSL, classes are split into two groups: classes whose data are available during the model's training stage ("seen" classes), and classes whose data only appear during inference ("unseen" classes). For both types, attribute vectors describing each class are available to facilitate knowledge transfer. During test time, samples may come from either class seen during training or new unseen classes. We report mean seen accuracy (mSA) and mean unseen accuracy (mUA), as well as the harmonic mean (mhM) of both as an overall metric; harmonic mean is considered preferable to simple arithmetic mean as an overall metric, as it prevents either term from dominating [13]. The harmonic mean (mhM) can be defined as:

$$mhM = \frac{2 \times mUA \times mSA}{mUA + mSA} \tag{14}$$

Note that some GZSL approaches (notably, generative ones) assume that the list of unseen classes and their attribute vectors are available during the training stage, even if their data are not; this inherently restricts these models to these known unseen classes. Conversely, our approach only requires the attributes of the seen classes. Also, in contrast to the continual GZSL settings described below, all seen classes are assumed available simultaneously during training.

3.1.1 Implementation Details

In the proposed model, Φ_a, Φ_s, and Φ_b are single-layer fully connected (fc) neural networks with ReLU, Sigmoid, and ReLU activation functions respectively. The dimension of each neural network (Φ_a, Φ_s, and Φ_b) is 2048. The self-gating output on the given attribute goes to another one-layer neural network of dimension $2048 \rightarrow 2048$ along with the BatchNorm layer. The output of this layer is considered the projected visual feature, and in the visual

[1] http://datasets.d2.mpi-inf.mpg.de/xian/xlsa17.zip

Table 1. The dataset and their split for the seen and unseen classes for the GZSL setting.

Dataset	Seen Classes	Train	Val	Unseen Classes (Test)	Attribute Dimension	Total Classes
AWA1 [7]	40	30	10	10	85	50
AWA2 [13]	40	30	10	10	85	50
CUB [12]	150	100	50	50	312	200
SUN [9]	645	500	145	72	102	717
aPY [1]	20	15	5	12	64	32

space, we measure the similarity by cosine distance. For all the datasets, the model is trained for the 200 epoch per task. For the inner loop, we use Adam [5] optimizer with a constant learning rate 0.0001. In the meta update, we use Adam optimizer with an initial learning rate of 0.001, and it decreases with the increase of the epoch at a rate of $(1 - current\_epoch/(total\_epoch - 1))$. We follow the same hyperparameter for all the datasets that shows the model's stability and applicability for the wide range of diverse datasets. The regressor network is also a 2048 dimensional fully connected layer, and MMR uses $\lambda = 5.0$.

3.2. Fixed Continual GZSL

The setting proposed by [4] divides all classes of the dataset into K subsets, each corresponding to a task. For task T_t, the first t of these subsets are considered the seen classes, while the rest are unseen; this results in the number of seen classes increasing with t while the number of unseen classes decreases. Over the span of $t = 1, ...K$, this simulates a scenario where we eventually "collect" labeled data for previously unseen classes. Note that in contrast to the typical GZSL setting, only data from the t^{th} subset are available; previous training data are assumed inaccessible. The goal is to learn from this newly "collected" data without experiencing catastrophic forgetting. As in GZSL, we report mSA, mUA, and mH, but at the end of $K-1$ tasks:

$$mSA_F = \frac{1}{K-1}\sum_{i=1}^{K-1} \text{Acc}(\mathcal{D}_{ts}^{i}(c_{\leq i}^{s}), \mathcal{A}(c_{\leq i}^{s})) \quad (15)$$

$$mUA_F = \frac{1}{K-1}\sum_{i=1}^{K-1} \text{Acc}(\mathcal{D}_{ts}^{i}(c_{i}^{u}), \mathcal{A}(c_{i}^{u})) \quad (16)$$

$$mhM_F = \frac{1}{K-1}\sum_{i=1}^{K-1} \mathcal{H}(\mathcal{D}_{ts}^{i}(c_{\leq i}^{s}), \mathcal{D}_{ts}^{i}(c_{i}^{u}), \mathcal{A}) \quad (17)$$

where Acc represents per class accuracy, $\mathcal{D}_{ts}^{i}(c_{\leq i}^{s})$ and $\mathcal{A}(c_{\leq t}^{s})$ are the seen class test data and attribute vectors respectively during the i^{th} task. Similarly $\mathcal{D}_{ts}^{i}(c_{i}^{u})$ and $\mathcal{A}(c_{i}^{u})$ represents the unseen class test data and attribute vectors during the i^{th} task. $\mathcal{H}$ is the harmonic mean of the accuracies obtained on $\mathcal{D}_{ts}^{i}(c_{\leq i}^{s})$ and $\mathcal{D}_{ts}^{i}(c_{i}^{u})$. We calculate the metric up to task $K-1$, as there are no unseen classes for task K, resulting in standard supervised continual learning.

3.3. Dynamic Continual GZSL

While it's not unreasonable that previously unseen class may become seen in the future, the above fixed continual GZSL evaluation protocol assumes that all unseen classes and attributes are set from the beginning, which may be unrealistic. An alternative framing of continual GZSL is one in which each task consists of its own disjoint set of seen and unseen classes, as proposed by [2]. Such a formulation does not require all attributes to be known *a priori*, allowing the model to continue accommodating an unbounded number of classes. As such, in contrast to the fixed continual GZSL, the number of seen and unseen classes both increase with t. As with the other settings, we report mSA, mUA and mH:

$$mSA_D = \frac{1}{K}\sum_{i=1}^{K} \text{Acc}(\mathcal{D}_{ts}^{i}(c_{\leq i}^{s}), \mathcal{A}(c_{\leq i}^{s})) \quad (18)$$

$$mUA_D = \frac{1}{K}\sum_{i=1}^{K} \text{Acc}(\mathcal{D}_{ts}^{i}(c_{\leq i}^{u}), \mathcal{A}(c_{\leq i}^{u})) \quad (19)$$

$$mhM_D = \frac{1}{K}\sum_{i=1}^{K} \mathcal{H}(\mathcal{D}_{ts}^{i}(c_{\leq i}^{s}), \mathcal{D}_{ts}^{i}(c_{\leq i}^{u}), \mathcal{A}) \quad (20)$$

where Acc represents per class accuracy, $\mathcal{D}_{ts}^{i}(c_{\leq i}^{s})$ and $\mathcal{A}(c_{\leq t}^{s})$ are the seen class test data and attribute vectors during i^{th} task. Similarly $\mathcal{D}_{ts}^{i}(c_{\leq i}^{u})$ and $\mathcal{A}(c_{\leq i}^{u})$ represents the unseen class test data and attribute vector during the i^{th} task. Detailed splits of the seen and unseen class samples for each task are given in the supplementary material.

3.3.1 Task Details

The AWA1 and AWA2 datasets contain 50 classes, which are divided into five tasks of ten classes each. We divide 717 classes of the SUN dataset into 15 tasks; the first three tasks contain 47 classes, and the remainder with 48 classes each. The CUB dataset contains 200 classes; we divide all classes into 20 tasks of ten classes each. The aPY dataset contains 32 classes; we divide the dataset into four tasks with eight classes in each task. The reservoir sample B for the AWA1, AWA2, CUB, SUN and aPY are $25, 25, 10, 5$ and 25 respectively.

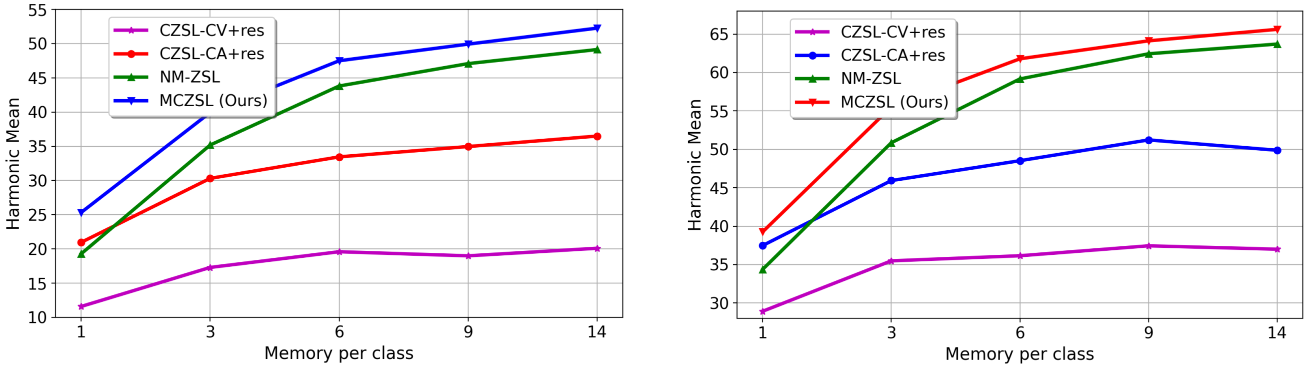

Figure 1. Model performance vs. Memory growth for continual GZSL on the CUB dataset, **Left:** Represents the Fixed Continual GZSL setting described in Section 3.2, **Right:** Represents the Dynamic Continual GZSL setting described in Section 3.3.

4. Ablation Studies

We conduct extensive ablation studies on the proposed model's different components, observing that each of the proposed components play a critical role. We show the effects of different components on the AWA1 and CUB datasets in the fixed continual GZSL setting, with more ablation studies for dynamic continual GZSL in the supplementary material.

4.1. Reservoir size vs Performance

To overcome catastrophic forgetting, the model uses a constant-size reservoir [8] to store previous task samples; with more tasks, the number of samples per class decreases. The reservoir size plays a crucial role in model performance. Figure 1, we evaluate the model's performance for both fixed and dynamic continual GZSL. We observe that for different reservoir sizes $\{1, 3, 6, 9, 14\} \times \#classes$, the proposed model shows consistently better results than recent models. For fixed and dynamic continual GZSL, $\#classes$ is $S + U$ and S, respectively.

4.2. t-SNE Visualization

Figure 2 shows the t-SNE plot for the AWA2 dataset. The attributes are projected to the visual space, and in the visual space, we do the t-SNE plot for all the samples. Here we observe that the projected attribute features closely correspond to the visual data space.

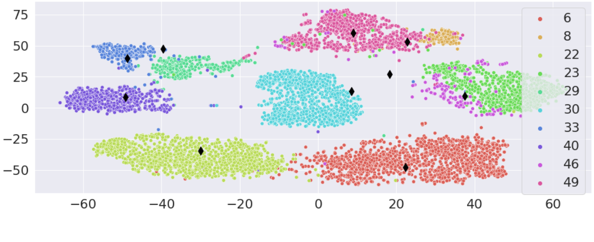

Figure 2. t-SNE plot for the unseen classes AWA2 dataset

Table 2. GZSL result when incorporating the with generated samples from MZSL [11]

	AWA1			CUB		
	mSA	mUA	mH	mSA	mUA	mH
MAIN	77.9	71.9	74.8	58.7	65.9	62.1
MAIN+MZSL [b]	75.3	70.1	72.6	57.6	61.7	59.5

4.3. Incorporating Generative model in the proposed approach

As we know that the generative model shows a promising result for the GZSL setting. Here a question arises if we combined the generative and discriminative approaches, how will the model behave? To answer the above question, we perform the experiment where the generated samples of the unseen classes are also incorporated into the model during training. We observe that doing so would result in similar disadvantages as our generative baselines: slower training and reduced flexibility. The result is shown in Table 2 where we use MZSL [11] model to generate the unseen class samples. We observe that we still surpass all baselines with generated samples, but it does not help either. We suspect this is due to a mismatch between real and generated samples.

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] Chandan Gautam, Sethupathy Parameswaran, Ashish Mishra, and Suresh Sundaram. Generalized continual zero-shot learning. *arXiv preprint arXiv:2011.08508*, 2020.

[3] Chandan Gautam, Sethupathy Parameswaran, Ashish Mishra, and Suresh Sundaram. Generative replay-based continual zero-shot learning. *arXiv preprint arXiv:2101.08894*, 2021.

[4] Skorokhodov Ivan and Elhoseiny Mohamed. Class normalization for zero-shot learning. In *International Conference on Learning Representations*, 2021.

[5] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] Hari Chandana Kuchibhotla, Sumitra S Malagi, Shivam Chandhok, and Vineeth N Balasubramanian. Unseen classes at a later time? no problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9245–9254, 2022.

[7] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009.

[8] David Lopez-Paz, Ranzato, Marc'Aurelio, and D Dummy. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476, 2017.

[9] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2751–2758. IEEE, 2012.

[10] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, pages 211–252, 2015.

[11] Vinay Verma, Kumar, Dhanajit Brahma, and Piyush Rai. A meta-learning framework for generalized zero-shot learning. *Association for the Advancement of Artificial Intelligence*, 2020.

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

[13] 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.