mathematics

MDPI

Article

Residual-Prototype Generating Network for Generalized Zero-Shot Learning

Zeqing Zhang [1,2,†], Xiaofan Li [2,†], Tai Ma [1], Zuodong Gao [2], Cuihua Li [2] and Weiwei Lin [3,*]

[1] School of Earth Sciences and Engineering, West Yunnan University of Applied Sciences, Dali 671000, China
[2] School of Informatics, Xiamen University, Xiamen 361000, China
[3] School of Big Data and Artificial Intelligence, Fujian Polytechnic Normal University, Fuqing 350300, China
* Correspondence: linww_cn@hotmail.com or linww@fpnu.edu.cn
† These authors contributed equally to this work.

Abstract: Conventional zero-shot learning aims to train a classifier on a training set (seen classes) to recognize instances of novel classes (unseen classes) by class-level semantic attributes. In generalized zero-shot learning (GZSL), the classifier needs to recognize both seen and unseen classes, which is a problem of extreme data imbalance. To solve this problem, feature generative methods have been proposed to make up for the lack of unseen classes. Current generative methods use class semantic attributes as the cues for synthetic visual features, which can be considered mapping of the semantic attribute to visual features. However, this mapping cannot effectively transfer knowledge learned from seen classes to unseen classes because the information in the semantic attributes and the information in visual features are asymmetric: semantic attributes contain key category description information, while visual features consist of visual information that cannot be represented by semantics. To this end, we propose a residual-prototype-generating network (RPGN) for GZSL that extracts the residual visual features from original visual features by an encoder–decoder and synthesizes the prototype visual features associated with semantic attributes by a disentangle regressor. Experimental results show that the proposed method achieves competitive results on four GZSL benchmark datasets with significant gains.

Keywords: deep learning; object recognition; generalized zero-shot learning; generative adversarial network

MSC: 68T07

Citation: Zhang, Z.; Li, X.; Ma, T.; Gao, Z.; Li, C.; Lin, W. Residual-Prototype-Generating Network for Generalized Zero-Shot Learning. *Mathematics* **2022**, *10*, 3587. https://doi.org/10.3390/math10193587

Academic Editors: Xiangtao Zheng, Jinchang Ren and Ling Wang

Received: 24 August 2022
Accepted: 22 September 2022
Published: 1 October 2022

1. Introduction

The emergence of deep convolutional networks [1–4] has greatly developed image recognition. However, the realistic category distribution is imbalanced, as some categories have few or even no samples for training, which severely limits the performance of networks on unseen classes. Zero-shot learning [5] is proposed to solve this problem by recognizing novel classes (unseen classes) solely through class-level semantic attributes and knowledge learned from the existing training set (seen classes). Conventional zero-shot learning has the strong assumption that all test samples are from unseen classes. In adaptation to realistic scenarios, generalized zero-shot learning (GZSL) [6,7] has been proposed, where seen classes and semantic attributes are used to train the model, and both seen and unseen classes participate in the test.

The current mainstream solution to GZSL is feature generative methods [8–11] that use semantic attributes as the cues to train a visual feature generator. The trained feature generator can be used to synthesize unseen-class visual features by the attributes of unseen classes. The synthetic unseen-class features and existing seen-class features are used to train a softmax classifier in a supervised way. The synthetic features make up for the lack of unseen classes and effectively avoid overfitting on seen classes. The generative methods use only semantic

attributes as cues for synthesizing visual features, which can be considered attribute-to-feature mapping. However, the visual features contain richer information than the semantic attributes, thus the semantic attributes can only represent visual features that are highly relevant to the category. Current generative methods concatenate Gaussian noises on the attributes to balance the gap between the attributes and the features, but the introduction of a large amount of noise seriously affects the mining of attributes, which restricts the attribute-to-feature inference of the generative model.

Through the discussion above, a residual-prototype-generating network (RPGN) is proposed, which consists of prototype generating and residual feature generating. Specifically, considering that different dimensions of semantic attributes represent different category information, such as white, furry, paws, etc., we propose a disentangle regressor that first chunks the semantic attributes and regresses them separately and then regresses the visual feature prototype. Through the disentangling operation, the visual feature prototype can fully explore the different information in the attribute and has excellent category differentiation. Meanwhile, to ensure consistency between feature prototypes and semantic attributes, the feature prototypes are reverted to the original semantic attributes as the reconstruction constraint. In addition, we propose a residual variational auto-encoder (rVAE) to extract the part of visual features (residual features) that cannot be represented by semantic attributes from the original visual features. The residual features extracted from original features have realistic and rich visual features and do not affect the mining of attribute information. To finish, the feature prototypes and the residual features are added to obtain the synthesized visual features, the category of which is determined by the category to which the prototype belongs, and the residuals used to synthesize the features can be synthesized with any visual features. This residual-prototype generation ensures that the feature prototypes contain attribute-associated information, and the residual features contain only attribute-independent information, which is indeed indispensable for constructing a real visual feature.

The work and contributions in this paper are threefold:

- We propose a disentangle regressor to generate class prototypes by first chunking the attributes and regressing them separately for fully mining visual information.
- We design a residual-prototype-generating network (RPGN) that can synthesize diverse unseen-class features with the help of real seen-class features, greatly enhancing the diversity of synthesized features.
- The proposed method achieves state-of-the-art (SOTA) results on four public GZSL datasets.

2. Related Work

2.1. Conventional Zero-Shot Learning

Zero-shot learning (ZSL) relies on class-level semantic descriptions or features, such as semantic attributes and word vectors, which are used to transfer models from seen classes to unseen classes. Early ZSL research work focused on traditional ZSL problems, in which semantic embedding was the most important approach. Semantic embedding methods can learn to embed visual features into semantic space. By doing this, visual features and semantic features are located in the same space, and ZSL classification can be accomplished by searching for the nearest semantic descriptor.

2.2. Generalized Zero-Shot Learning

Currently, generative approaches dominate in GZSL [6,7]; these exploit existing adversarial generative networks (GAN) [12–14] or variational auto-encoders (VAE) [15,16] to synthesize visual features from class-level semantic attributes and random noise. Examples such as f-CLSWGAN [8], cycle-UWGAN [11] and LisGAN [10] introduce the Wasserstein generative adversarial network (WGAN) [17] paired with a pre-trained classifier to synthesize visual features for unseen classes, transforming the GZSL task into a fully supervised classification issue. RFF [18] combines the traditional projection

method and GAN to initially map visual features to a new redundancy-free feature space and then to judge the veracity of the mapped features. To enhance the performance of the generator, some works [9,19–21] formulate GAN into the variational auto-encoder (VAE) model to fit the class-specific latent distribution and highly discriminative feature representations. The combination of GAN and VAE greatly enhances the performance of the generative model in GZSL and becomes the benchmark method for many new methods [22,23].

To enhance the generalization of the model from seen classes to unseen classes, some studies introduce meta-learning. Meta-learning methods for GZSL [24–26] simulate the setting of zero-shot learning in the training stage so that the model can effectively learn the ability to transfer knowledge. ZSML [25] first introduces meta-learning into GZSL and combines meta-learning with GAN, which gives the generator better knowledge transfer capability. E-PGN [24] first introduces an episode-based paradigm in the training phase; this divides the training set into pseudo-multiple zero-shot learning subsets and then takes these subsets as multiple tasks to train the model. By training on multiple tasks, the model can effectively transfer the knowledge learned from the seen classes to the task of the unseen classes. Furthermore, to solve the problem of task inconsistency in the meta-learning model, TGMZ [26] proposed a task alignment module to make the tasks of seen classes and unseen classes more similar.

3. Method

3.1. Notations and Definitions

In the GZSL task, we define the dataset of seen classes as $\mathcal{S} = \{(x^s, y^s, a^s) \mid x^s \in \mathcal{X}^s, y^s \in \mathcal{Y}^s, a^s \in \mathcal{A}^s\}$, where $\mathcal{X}^s$, $\mathcal{Y}^s$ and $\mathcal{A}^s$ represent the set of visual features, labels and attributes of seen classes, respectively. Similarly, the unseen class dataset is denoted as $\mathcal{U} = \{(x^u, y^u, a^u) \mid x^u \in \mathcal{X}^u, y^u \in \mathcal{Y}^u, a^u \in \mathcal{A}^u\}$, where $\mathcal{X}^u$, $\mathcal{Y}^u$ and $\mathcal{A}^u$ denote the set of visual features, labels and attributes of unseen classes, respectively. Note that the label set of seen and unseen classes do not intersect, which is $\mathcal{Y}^s \cap \mathcal{Y}^u = \varnothing$. The main goal of the GZSL task is to train a classifier $f : \mathcal{X}^s \cup \mathcal{X}^u \rightarrow \mathcal{Y}^s \cup \mathcal{Y}^u$ that can recognize both seen and unseen classes by only using the seen-class dataset ($\mathcal{S}$) and attributes of unseen classes ($\mathcal{A}^u$) for training.

3.2. Overview

The framework of the proposed approach, called residual-prototype-generating network (RPGN), is shown in Figure 1 and contains five modules: ResNet-101 [4] as a backbone, an encoder (*En*), a decoder (*De*), a disentangle regressor (*Rd*), an attribute regressor (*Ra*), and a discriminator (*D*).

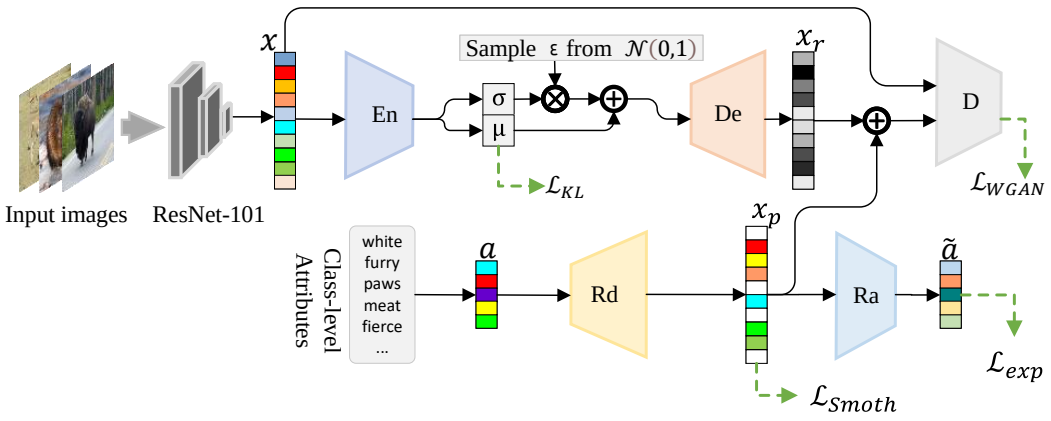

Figure 1. The overall framework of our proposed residual-prototype-generating network (RPGN).

The backbone is ResNet-101 pre-trained on ImageNet to extract visual features of images, and its parameters are fixed and no longer fine-tuned in any of the experiments.

The main function of disentangling regressor Rd is to generate a prototype feature of a certain class based on attributes, and its structure is a block connection multilayer perceptron (MLP) (see Figure 2 for details) for capturing complex semantic relationships between attributes. To contain richer information on attributes, the generated prototype feature reconstructs the attributes through an attribute regressor (Ra).

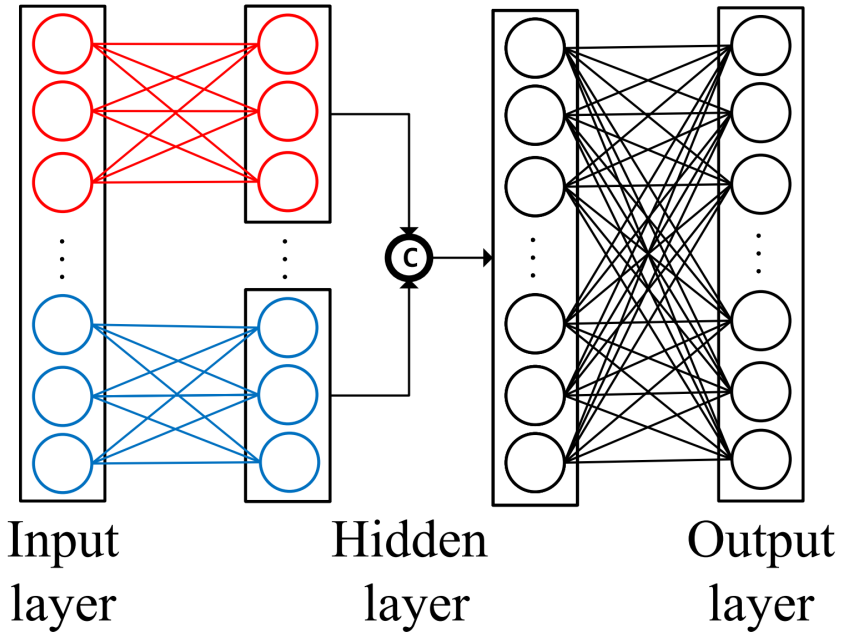

Figure 2. Illustration of the proposed disentangle regressor; C denotes concatenate operation.

The encoder (En) and decoder (De) form the proposed residual variational auto-encoder (rVAE), which generates residual features. Unlike the traditional variational auto-encoder that synthesizes features directly, the proposed rVAE is used to extract semantically irrelevant visual feature residuals from real features. The prototype features synthesized by the disentangle regressor (Rd) only have semantic information and do not contain other information that constitutes visual features; thus the residual features can be used to compensate for the missing prototype feature information. Further, through generating prototype features and residual features, RPGN can make the final synthesized visual features contain more semantic information and be more realistic.

The discriminator (D) mainly distinguishes between real features and synthetic features and makes the distribution of synthetic features converge to the distribution of real features through adversarial learning loss. The attribute regressor reconstructs the prototypes back to the attributes, ensuring that the prototypes and attributes are strongly correlated.

3.3. Prototype Generation

The prototype feature is generated through the proposed disentangle regressor (Rd), which is illustrated in Figure 2. The reason for this design is that there are strong dependencies between attributes; for example, the presence of a bird's beak in an attribute necessarily results in the presence of a bird's claw, which leads to coupling between attributes. To solve this problem, the attributes (a) are divided into K blocks $(a^1, a^2, \cdots, a^K)$ randomly to reduce the dependencies between attributes. Each attribute block (a^i) is input to the hidden layer (f_i) to get the result, and then all of them are concatenated together and input to the output layer (f_o) to get the prototype feature x_p with the following equation:

$$x_p = Rd(a) = f_o([f_1(a^1), f_2(a^2), \cdots, f_k(a^k)]), \tag{1}$$

where a^i denotes the attribute of the i-th block, and $[a^1, a^2, \cdots, a^K] = a$.

To make the prototype contain the common features of a certain class, all the visual features (x) of this class are used to constrain the prototype (x_p). Usually, L2 loss is used to achieve this, but each class has only one prototype, which will inevitably lead to some features of this class being greatly different from the prototype, causing too much L2 loss; that is, it is easy to have a gradient explosion. To avoid this problem, smooth L1 loss [27] is used here, which is as follows:

$$\mathcal{L}_{Smoth} = \begin{cases} 0.5\mathbb{E}_x[\| x_p - x \|_2^2 / \alpha], & \| x_p^s - x^s \|_2^2 < \alpha, \\ \mathbb{E}_x[\| x_p - x \|_1] - 0.5\alpha, & otherwise, \end{cases} \tag{2}$$

where α is a hyper-parameter and is set to 1 in all the experiments according to the suggestion of [27].

To further constrain the prototype features to be as relevant as possible to the attributes and to contain all the attribute information, the prototype is reconstructed back to the attributes, and an exponential loss function is used to make the projected attributes to be as similar as possible to the real attributes:

$$\mathcal{L}_{exp} = \mathbb{E}_{x_p}[e^{-R(x_p)\cdot a}]. \tag{3}$$

The purpose of exponential loss is to produce a high dot-product similarity between the reconstructed attributes and the real ones.

3.4. Residual Generation

Residual generation is implemented through the proposed residual variational autoencoder (rVAE), which first encodes visual feature (x) as a hidden variable (denoted as h). Here, we use the re-parameterization trick in VAE [28] to represent h as $h = \mu + \sigma * z$, where $[\sigma, \mu] = En(x), z \sim \mathcal{N}(0,1)$, and the distribution of h is consistent with the standard normal distribution through the following Kullback–Leibler divergence loss:

$$\mathcal{L}_{KL} = \frac{1}{2}\mathbb{E}_{\mu,\sigma}[\sum_{i=1}^{d}(\mu_i^2 + \sigma_i^2 - log(\sigma_i^2) - 1)], \tag{4}$$

where σ_i, μ_i are the i-th dimension values of σ and μ, respectively.

The latent variables are used to decode the residual features with the following equation:

$$x_r = De(h) = De(\mu + \sigma * z), \tag{5}$$

where z is Gaussian noise; that is, $z \sim \mathcal{N}(0,1)$. In the absence of confusion, the process of synthesizing residual features by the residual generator is abbreviated as $x_r = G_R(x,z)$. Here, the final residual features are not expected to be the same as the initial visual features as in traditional VAE, but are expected to contain the attribute-independent part of a visual feature, which is precisely the part that makes the final synthesized visual features more realistic and more diverse. The information of residual features cannot be obtained by traditional generators through random noise alone, and the proposed method greatly avoids the problem of missing the diversity of synthetic features by traditional generators.

3.5. Generative Adversarial Network

The final synthetic features combine the results of the prototype generator and the residual generator as follows:

$$\hat{x} = x_p + x_r. \tag{6}$$

Note that the residual features x_r are synthesized using the real features of any seen classes, not necessarily the same class as the prototype features x_p, and the final synthesized features x are determined by the class to which the prototype belongs. This way of synthesizing residual features without using visual features of the same class can significantly improve residual generation to capture attribute-independent feature information, which can greatly enhance the authenticity of the synthesized features. For convenience, the

overall generation process is denoted as $\hat{x} = G(a, x, z)$, where x is the real feature used to extract the residual feature, z is sampled from $\mathcal{N}(0, 1)$ for re-parameterization, and a is the attribute vector to control the class of synthesized feature $\hat{x}$.

To improve the authenticity of the synthetic features and to make their distribution closer to that of the real features, the generative adversarial loss is introduced as:

$$\begin{aligned} \mathcal{L}_D = & \mathbb{E}[D(x, a)] - \mathbb{E}[D(\hat{x}, a)] \\ & - \beta \mathbb{E}[(||\nabla_{\dot{x}} D(\dot{x}, a)||_2 - 1)^2] \end{aligned} \tag{7}$$

where β is a hyper-parameter. The last term is the Wasserstein loss [17] by enforcing the Lipschitz constraint [29], where $\dot{x} = \gamma x + (1 - \gamma)\hat{x}$ with $\gamma \sim U(0, 1)$. As suggested in [17], we fix $\beta = 10$.

In summary, the total objective loss is as follows:

$$\max_{D} \min_{En, Dn, Rd, Ra} \mathcal{L} = \mathcal{L}_D + \mathcal{L}_{KL} + \mathcal{L}_{Smoth} + \lambda \mathcal{L}_{exp}, \tag{8}$$

where λ is a hyper-parameter to control the effect of attribute reconstruction.

3.6. Classification

In the classification stage, the goal is to train a classifier capable of classifying both seen and unseen classes. To compensate for the lack of unseen class data, a synthetic unseen class dataset is forged through feature generation. Specifically, given the a_y^u of any unseen class $y \in \mathcal{Y}^u$, and with sampling noise z and seen-class feature x^s, the visual features of unseen classes can be synthesized through $\hat{x} = G(a^u, x^s, z)$. After repeating this feature-generation process for every unseen class, a synthetic unseen dataset $\bar{\mathcal{U}}$ is obtained. As the last step, synthetic unseen classes $\bar{\mathcal{U}}$ and real seen classes $\mathcal{S}$ are used to train a classifier in a supervised way.

4. Experimental Results

4.1. Experimental Setup

Datasets. The proposed method is evaluated on four benchmark datasets: Animals with Attributes (AWA1) [30], renewed Animals with Attributes (AWA2) [6], Caltech USCD Birds-2011 (CUB) [31] and Oxford Flowers (FLO) [32]. We use the category splitting strategy in [6]. Concretely, AWA1 and AWA2 are two coarse-grained datasets that consist of different visual images from the same 50 animal classes; each class is labeled with 85-dimensional category attributes. Forty classes in AWA1 and 2 are split into seen classes, and the other 10 classes are split into unseen classes. CUB is a fine-grained dataset with 11,788 images from 200 different classes of birds with 1024-dimensional attributes extracted by CNN-RNN [33]. A total of 150 classes in CUB are split into seen classes, and the other 50 classes are split into unseen classes. FLO is a fine-grained dataset that consists of 8189 images from 102 flower classes with 1024-dimensional attributes extracted by CNN-RNN [33], similar to CUB. A total of 82 classes in FLO are split into visible classes, and another 20 classes are split into unseen classes. A summary of the datasets is given in Table 1.

Table 1. Statistics of five benchmark datasets. From left to right, the columns show the dataset name, the class attribute dimensionality $\mathcal{A}$, the number of seen classes $\mathcal{Y}^s$ and unseen classes $\mathcal{Y}^u$, all training instances $\mathcal{S}^{tr}$, and testing seen instances $\mathcal{S}^{te}$ and unseen instances $\mathcal{U}^{te}$.

Dataset	$\mathcal{A}$	$\mathcal{Y}^s$	$\mathcal{Y}^u$	$\mathcal{S}^{tr}$	$\mathcal{S}^{te}$	$\mathcal{U}^{te}$
CUB	1024	150	50	11,788	1764	2967
AWA1	85	40	10	30,475	5685	4958
AWA2	85	40	10	37,322	5882	7913
FLO	1024	82	20	5394	1640	1155

Evaluation Protocol. The proposed method is evaluated in terms of the average per-class Top-1 accuracy (ACC). In addition, we follow the protocol proposed in [6] for the GZSL task, in which the ACCs of both seen class S and unseen class U are calculated, as well as the harmonic mean $H = 2 \times S \times U/(S + U)$.

Implementation Details. Following [6], we use the 2048-dimensional global features extracted by ResNet-101 [4], which is pre-trained on ImageNet for visual features. As a pre-processing step, we also normalize the visual features such as that of f-CLSGAN [8]. *En*, *De*, *D* and *Ra* all adopt a three-layer multilayer perceptron (MLP), which employs 4096 units in the hidden layer. Meanwhile, they all apply LeakReLU as the activation function on the hidden layer. The difference is that the *De*, *Rd* and *Ra* apply ReLU [34,35] as the activation function of the output layer, while *D* and *En* do not use the activation function on the output layer. For all datasets, *En*, *De*, *D*, *Rd* and *Ra* are optimized by Adam [36] with learning rate 0.0001, batch size 512, $\beta_1 = 0.999$ and $\beta_2 = 0.5$.

4.2. Comparison with SOTA Methods

Results on public zero-shot learning datasets are reported in Table 2. RPGN outperforms all generative and non-generative methods on AWA2, a coarse-grained dataset, and FLO, a dataset with non-manually labeled attributes, especially on the FLO dataset, where the proposed method is only 3% away from reaching 80% accuracy. On AWA1, RPGN outperforms all the generative methods by 1.1% over the second place CE-GZSL [37], which is a very big improvement. On CUB, a fine-grained dataset, RPGN is below CE-GZSL [37] in the generative method, but it is much better than the algorithm on other datasets, which shows that the proposed method can be applied to all datasets and does not cause significant performance degradation due to changes in the dataset. In addition, the unseen class accuracy U of RPG is the highest on AWA2 compared with the generative method; our method is higher than TGMZ [26] by 4.2%, which is a good indication that our model is able to synthesize more truly diverse unseen-class features, and although the unseen-class accuracy of our method is not the highest on FLO, we have the highest seen-class accuracy, which also shows that our synthesized unseen-class features are not strongly biased towards the seen class; therefore, they do not affect the accuracy of the seen class too much.

Table 2. Comparative results with other methods. Red font and blue font denote the highest and the second highest results, respectively.

	Methods	CUB			AWA1			AWA2			FLO		
		U	***S***	***H***	***U***	***S***	***H***	***U***	***S***	***H***	***U***	***S***	***H***
Non-Generative	SE-ZSL [38]	41.5	53.3	46.7	56.3	67.8	61.5	58.3	68.1	62.8	-	-	-
	ABP [39]	47.0	54.8	50.6	57.3	67.1	61.8	55.3	72.6	62.6	-	-	-
	COSMO [40]	44.4	57.8	50.2	52.8	80.0	63.6	-	-	-	59.6	81.4	68.8
	ZSML [25]	60.0	52.1	55.7	57.4	71.1	63.5	58.9	74.6	65.8	-	-	-
	DAZLE [41]	59.6	56.7	58.1	-	-	-	75.7	60.3	67.1	-	-	-
	DVBE [42]	53.2	60.2	56.5	-	-	-	63.6	70.8	67.0	-	-	-
	E-PGN [24]	52.0	61.1	56.2	62.1	83.4	71.2	52.6	83.5	64.6	71.5	82.2	76.5
	GCM-CF [43]	61.0	59.7	60.3	-	-	-	60.4	75.1	67.0	-	-	-
	SR2E [44]	61.6	70.6	65.8	-	-	-	58.0	80.7	67.5	-	-	-
	AGZSL [45]	41.4	49.7	45.2	-	-	-	65.1	78.9	71.3	-	-	-
	IPN [46]	73.8	60.2	66.3	-	-	-	79.2	67.5	72.9	-	-	-

Table 2. *Cont.*

	Methods	CUB			AWA1			AWA2			FLO		
		U	***S***	***H***	***U***	***S***	***H***	***U***	***S***	***H***	***U***	***S***	***H***
Generative	GAZSL [47]	31.7	61.3	41.8	29.6	84.2	43.8	35.4	86.9	50.3	28.1	77.4	41.2
	f-CLSGAN [8]	43.7	57.7	49.7	57.9	61.4	59.6	56.1	65.5	60.4	59.0	73.9	65.6
	cycle-UWGAN [11]	45.7	61.0	52.3	56.9	64.0	60.2	-	-	-	59.2	72.5	65.1
	LisGAN [10]	46.5	57.9	51.6	52.6	76.3	62.3	47.0	77.6	58.5	57.7	83.8	68.3
	f-VAEGAN [9]	48.4	60.1	53.6	-	-	-	56.8	57.6	70.6	63.5	74.9	64.6
	CADA-VAE [19]	51.6	53.5	52.4	57.3	72.8	64.1	55.8	75.0	63.9	-	-	-
	DE-VAE [20]	52.5	56.3	54.3	59.6	76.1	66.9	58.8	78.9	67.4	-	-	-
	OCD-CVAE [21]	44.8	59.9	51.3	-	-	-	59.5	73.4	65.7	-	-	-
	RFF [18]	52.6	56.6	54.6	59.8	75.1	66.5	-	-	-	65.2	78.2	71.1
	FREE [23]	55.7	59.9	57.7	62.9	69.4	66.0	60.4	75.4	67.1	67.4	84.5	75.0
	HSVA [48]	52.7	58.3	55.3	59.3	76.6	66.8	56.7	79.8	66.3	-	-	-
	CE-GZSL [37]	63.9	66.8	65.3	65.3	73.4	69.1	63.1	78.6	70.0	69.0	78.7	73.5
	TGMZ [26]	60.3	56.8	58.5	65.1	69.4	67.2	64.1	77.3	70.1	-	-	-
	RPGN(Ours)	61.0	62.2	61.6	64.0	77.8	70.2	68.3	78.8	73.2	68.2	88.9	77.0

As for the reason why our method is not in the leading position on CUB, a fine-grained dataset, but is superior on AWA1, AWA2 and coarse-grained datasets, the analysis is as follows. Other generative methods only use attributes and Gaussian noise to synthesize unseen class data. If the attributes are coarse-grained and provide less information, the Gaussian noise alone cannot provide other information about a visual feature; therefore, the unseen-class features synthesized by other generative methods are bound to face problems of poor realism and lack of diversity. In contrast, our RPGN uses the residuals generated by real seen-class features. Due to the authenticity and richness of seen-class features, the generated residuals must also contain a large amount of information on real features, which can make up for the lack of information provided by attributes. As for CUB, a fine-grained dataset, the attributes already provide sufficient information; therefore, other generative methods obtain better performance. In summary, our method is more suitable for coarse-grained datasets than other methods. In addition, it is observed that many non-generative methods [44,46] can obtain very good performance on CUB, which is also because the attributes of CUB contain enough information that extra synthetic unseen-class features are not necessary to compensate for the absence of the unseen class, and the model can infer the relationship between features and attributes to complete zero-shot recognition.

4.3. Ablation Study

We perform ablation experiments to determine whether or not to use the following key components: (a) residual variational auto-encoder to generate residual features; (b) the disentangle regressor to synthesize the prototype; and (c) smooth L1 loss [27]. Results of the ablation study are reported in Table 3.

Residual feature generation: If only Gaussian noise is used to synthesize the residual features, this synthesis is similar to other generative methods, but the authenticity and diversity of the visual features synthesized by relying only on random noise to obtain information other than attributes of a visual feature are deficient because random noise does not provide as much information as the real visual features. In contrast, our method synthesizes residual features by random noise and real seen-class features; these residual features contain information other than the attributes in real visual features, and the richness and authenticity of the visual features synthesized with this residual feature method are guaranteed. Thus, comparing the first and second columns of Table 3 after using our method to synthesize visual features shows improvements of 5.0%, 6.2% and 6.1% on CUB, AWA2 and FLO, respectively. Comparing the fifth and eighth rows reveals

that removing the innovation reduces performance by 3.6%, 4.0% and 5.2% on the three datasets, respectively, which fully illustrates the effectiveness of the innovation.

Table 3. Results of ablation study: usage of (a) real seen classes to synthesize residual features; (b) a multi-headed MLP to synthesize the prototype; and (c) smooth L1 loss.

			CUB			AWA2			FLO		
a	***b***	***c***	***U***	***S***	***H***	***U***	***S***	***H***	***U***	***S***	***H***
			50.5	52.6	51.5	57.4	66.8	61.7	60.5	74.0	66.6
✓			55.5	57.6	56.5	63.6	72.9	67.9	66.6	80.1	72.7
	✓		54.8	56.8	55.8	62.5	73.6	67.6	64.2	79.9	71.2
		✓	53.9	56.7	55.3	62.9	70.0	66.3	63.8	77.0	69.8
	✓	✓	57.8	58.2	58.0	64.6	74.4	69.2	65.0	80.3	71.8
✓		✓	59.4	60.2	59.8	67.8	76.5	71.9	67.8	85.5	75.6
✓	✓		60.0	61.1	60.5	68.0	77.0	72.2	66.9	87.8	75.9
✓	✓	✓	61.0	62.2	61.6	68.3	78.8	73.2	68.2	88.9	77.0

Disentangle regressor: If only ordinary MLP is used to capture the semantic relationships between attributes, there will be strong dependencies between some attributes and confusion will arise between them, and some attributes will even be useless. The proposed disentangle regressor first randomizes attributes to different blocks, reducing the probability of having attributes with strong dependency in the same block and thus achieving better mining of the semantic relationships between attributes. This enables the synthesized prototype to make full use of all the attributes. Comparing the first and third rows of Table 3 shows that the model improves by 4.3%, 5.9% and 5.6% for CUB, AWA2 and FLO, respectively, after using the multi-headed MLP. Comparing the sixth and eighth rows reveals that the model's performance decreases by 1.8%, 1.3% and 1.4% for the three datasets, respectively, after removing the innovation, which fully indicates that the more complex structure can better generate prototypes containing attribute information, thus improving the zero-shot recognition performance of the model.

Smooth L1 loss: If the traditional L2 loss is used to constrain the prototypes and training features, there are bound to be many features that are far different from the prototypes, for there is only one prototype for each class. This will result in larger loss and will potentially lead to gradient explosion. In fact, the prototypes are expected to represent the universal and common information of a class, so if some features are more different than other features of the same class, then such features should not provide too much reference to the prototype. L2 loss would give such features more attention, which is obviously unreasonable. Comparing the first and third rows of Table 3, the model improves by 3.3%, 4.6% and 3.2% on CUB, AWA2 and FLO, respectively, after using the smooth L1 loss, while comparing the seventh and eighth rows after removing the innovation, the model's performance decreases by 1.1%, 1.0% and 1.1% on the three datasets, respectively, which fully illustrates the effectiveness of the loss and shows that the loss is more helpful to synthesize better prototypes.

4.4. Hyper-Parameter Analysis

Hyper-parameter K**:** We evaluate the effect of K as shown in Figure 3. We found that the effect of K on the accuracy of the model is smaller in a certain range. On the AWA2 datasets, the model gets the best performance when K is four, but on FLO and CUB, the model gets the best performance when K is 8. This is because the attributes of FLO and CUB are 1024 dimensions, which is far more than the other datasets, so they should be divided into more groups for the attributes. It can be seen that the number of groups should be positively correlated with the dimensionality of the attributes, and the larger the dimensionality of the attributes, the more the number of groups should be appropriately increased.

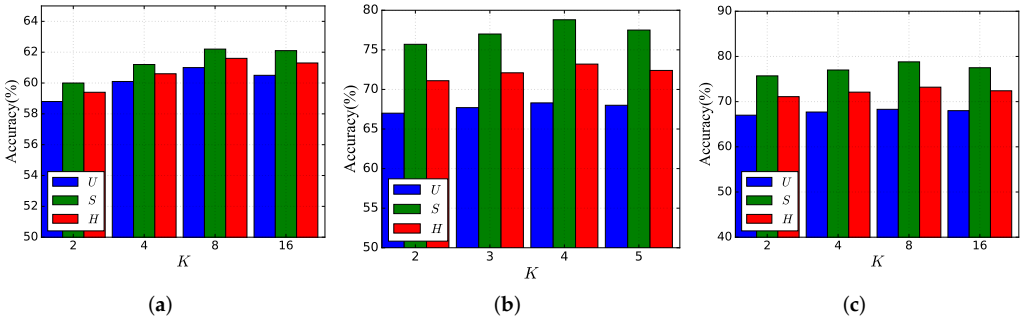

Figure 3. The effect of hyper-parameter K: (**a**) CUB; (**b**) AWA2; and (**c**) FLO.

Hyper-parameter λ: We evaluate the effect of λ as shown in Figure 4. This hyper-parameter controls the regression loss of the prototype projected back to the attribute space, and by the same token, the prototype is also controlled by the adversarial loss. If this hyper-parameter is too large, it will lead to a lower percentage of adversarial loss, which will lead to lower authenticity of the final synthesized features. If this hyper-parameter is too small, then it will lead to a lower correlation between the prototype and the attributes. Thus, λ is set to one to achieve the best results on all datasets.

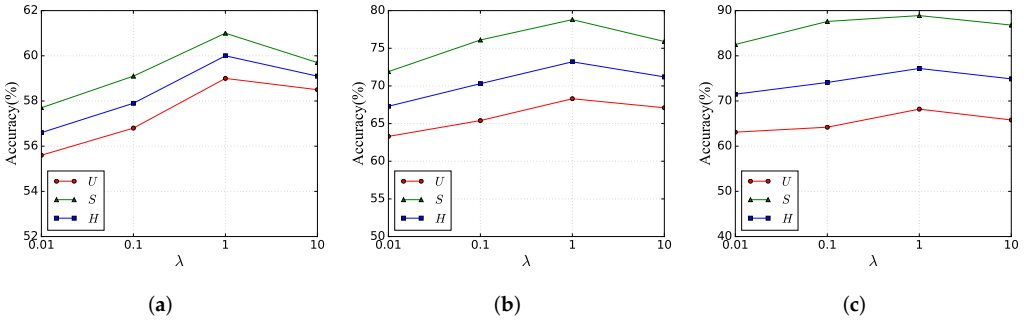

Figure 4. The effect of hyper-parameter λ: (**a**) CUB; (**b**) AWA2; and (**c**) FLO.

4.5. Zero-Shot Bai Character Recognition

In addition to the public datasets, we create a new zero-shot dataset: the Chinese and Bai Character (CBC) dataset. Bai script is an ancient script similar to Chinese script and is composed of 32 strokes, similar to Chinese (see Figure 5). Specifically, we take 503 Chinese characters as seen classes and 400 Bai characters as unseen classes and use the 32-stroke information of each character as attributes.

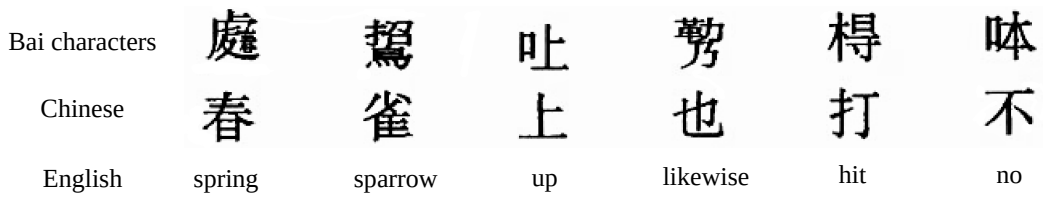

Figure 5. Excerpts of Chinese and Bai characters.

Results on CBC are reported in Table 4. We transplanted eight of the latest GZSL methods (ZSML [25], f-CLSGAN [8], LisGAN [10], f-VAEGAN [9], RFF [18], CE-GZSL [37] and FREE [23]) to the CBC dataset, and the best-performing RFF [18] has a recognition accuracy of 80.9% on the term of H. This is because these generative methods use Gaussian noise plus attribute-synthesized features; therefore, these synthesized features are less realistic and less diverse, while our RGPN uses residual features generated by real seen classes and prototype-synthesized features generated by attributes; therefore, our model synthesizes features that are more realistic and diverse. In addition, our method obtains

an accuracy of 79.4% on the unseen class and is 5.4% higher than FREE [23], which fully illustrates that the unseen-class features synthesized by our method are far better than other methods. Finally, on the index H, our method is also higher than the second place by 3.1%, which shows the superiority of our method. To be as fair as possible, the above eight GZSL methods all require careful hyper-parameter tuning to obtain comparable results.

Table 4. Comparative results with other methods on CBC. Red font and blue font denote the highest and the second highest results, respectively.

Methods	CBC		
	U	S	H
f-CLSGAN [8]	43.5	63.5	51.6
LisGAN [10]	40.9	72.7	52.3
f-VAEGAN [9]	42.7	71.1	53.4
ZSML [25]	53.4	67.7	59.7
RFF [18]	44.2	72.6	54.9
CE-GZSL [37]	52.2	70.4	59.9
FREE [23]	48.9	68.4	57.3
RPGN (Ours)	52.8	76.1	62.3

5. Conclusions

Current generative methods for GZSL are facing bottlenecks as a consequence of the fact that these methods use only Gaussian noise and attributes to synthesize unseen-class features. To get out of this predicament, we propose a novel residual-prototype adversarial generation network that can use real seen-class features to assist in synthesizing unseen-class features, greatly improving the richness and authenticity of unseen-class features. Experiments on multiple datasets, rich ablation experiments and hyper-parameter analysis experiments show that our method is comparable or superior to existing methods. Furthermore, we constructed a GZSL-compliant task with the Chinese and Bai character (CBC) dataset and applied eight SOTA methods to the CBC dataset to compare the performance.

Author Contributions: Conceptualization, Z.Z., X.L. and Z.G.; methodology, Z.Z. and X.L.; validation, Z.Z., X.L. and Z.G.; formal analysis, Z.Z. and X.L.; investigation, X.L. and Z.G.; resources, Z.Z. and T.M.; writing—original draft preparation, Z.Z. and X.L.; writing—review and editing, Z.G., C.L. and W.L.; visualization, X.L.; funding acquisition, T.M. All authors have read and agreed to the published version of the manuscript.

Funding: Project supported by the Science and Technology Planning Project of Yunnan Provincial Science and Technology Department (grant no. 2018FD058).

Data Availability Statement: Data will be provided upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
2. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
3. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition, 2015. Tech Report. *arXiv* **2014**, arxiv:1512.03385.
5. Lampert, C.H.; Nickisch, H.; Harmeling, S. Attribute-based classification for zero-shot visual object categorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 453–465. [CrossRef] [PubMed]
6. Xian, Y.; Lampert, C.H.; Schiele, B.; Akata, Z. Zero-shot learning—A comprehensive evaluation of the good, the bad and the ugly. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 2251–2265. [CrossRef] [PubMed]
7. Pourpanah, F.; Abdar, M.; Luo, Y.; Zhou, X.; Wang, R.; Lim, C.P.; Wang, X.Z.; Wu, Q.J. A review of generalized zero-shot learning methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, 1–20. [CrossRef] [PubMed]

8. Xian, Y.; Lorenz, T.; Schiele, B.; Akata, Z. Feature generating networks for zero-shot learning. In Proceedings of the Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 5542–5551.
9. Xian, Y.; Sharma, S.; Schiele, B.; Akata, Z. F-VAEGAN-D2: A feature generating framework for any-shot learning. In Proceedings of the CVPR, Long Beach, CA, USA, 16–20 June 2019; pp. 10275–10284.
10. Li, J.; Jing, M.; Lu, K.; Ding, Z.; Zhu, L.; Huang, Z. Leveraging the Invariant Side of Generative Zero-Shot Learning. In Proceedings of the CVPR, Long Beach, CA, USA, 16–20 June 2019; pp. 7402–7411.
11. Felix, R.; Kumar, B.G.V.; Reid, I.D.; Carneiro, G. Multi-modal cycle-consistent generalized zero-shot learning. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
12. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.C.; Bengio, Y. Generative adversarial nets. In Proceedings of the Advances in Neural Information Processing Systems 27 (NIPS 2014), Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
13. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arxiv:1411.1784.
14. Nguyen, T.D.; Le, T.; Vu, H.; Phung, D.Q. Dual discriminator generative adversarial nets. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; pp. 2670–2680.
15. Doersch, C. Tutorial on Variational Autoencoders. *arXiv* **2016**, arXiv:1606.05908.
16. Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. Beta-VAE: Learning basic visual concepts with a constrained variational framework. In Proceedings of the 5th International Conference on Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017.
17. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein generative adversarial networks. In Proceedings of the 34th International Conference on Machine Learning, ICML, PMLR 2017, Seoul, Korea, 15–17 November 2017; Volume 70, pp. 214–223.
18. Han, Z.; Fu, Z.; Yang, J. Learning the redundancy-free features for generalized zero-shot object recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 12862–12871.
19. Schonfeld, E.; Ebrahimi, S.; Sinha, S.; Darrell, T.; Akata, Z. Generalized zero- and few-shot learning via aligned variational autoencoders. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8247–8255.
20. Peirong, M.; Xiao, H. A variational autoencoder with deep embedding model for generalized zero-shot learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
21. Keshari, R.; Singh, R.; Vatsa, M. Generalized Zero-Shot Learning Via Over-Complete Distribution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.
22. Narayan, S.; Gupta, A.; Khan, F.S.; Snoek, C.G.; Shao, L. Latent embedding feedback and discriminative features for zero-shot classification. In Proceedings of the ECCV 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 479–495.
23. Chen, S.; Wang, W.; Xia, B.; Peng, Q.; You, X.; Zheng, F.; Shao, L. Free: Feature refinement for generalized zero-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11 October 2021; pp. 122–131.
24. Yu, Y.; Ji, Z.; Han, J.; Zhang, Z. Episode-based prototype generating network for zero-shot learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 14032–14041.
25. Verma, V.K.; Brahma, D.; Rai, P. A Meta-Learning Framework for Generalized Zero-Shot Learning. *arXiv* **2020**, arXiv:1909.04344.
26. Liu, Z.; Li, Y.; Yao, L.; Wang, X.; Long, G. Task aligned generative meta-learning for zero-shot learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 2–9 February 2021.
27. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
28. Caterini, A.L.; Doucet, A.; Sejdinovic, D. Hamiltonian variational auto-encoder. In Proceedings of the Advances in Neural Information Processing Systems 31 (NeurIPS 2018), Montréal, QC, Canada, 3–8 December 2018; pp. 8178–8188.
29. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A. Improved training of wasserstein GANs. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
30. Lampert, C.H.; Nickisch, H.; Harmeling, S. Learning to detect unseen object classes by between-class attribute transfer. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition CVPR, Miami, FL, USA, 20–25 June 2009; pp. 951–958.
31. Wah, C.; Branson, S.; Welinder, P.; Perona, P.; Belongie, S. *The Caltech-UCSD Birds-200-2011 Dataset*; Computation & Neural Systems Technical Report, 2010-001; California Institute of Technology: Pasadena, CA, USA, 2011.
32. Nilsback, M.E.; Zisserman, A. Automated flower classification over a large number of classes. In Proceedings of the 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, ICVGIP, Bhubaneswar, India, 16–19 December 2008; pp. 722–729.
33. Reed, S.; Akata, Z.; Lee, H.; Schiele, B. Learning deep representations of fine-grained visual descriptions. In Proceedings of the 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing CVPR, Bhubaneswar, India, 16–19 December 2008; pp. 49–58.
34. Agarap, A.F. Deep Learning using Rectified Linear Units (ReLU). *arXiv* **2018**, arXiv:1803.08375.
35. Maennel, H.; Bousquet, O.; Gelly, S. Gradient Descent Quantizes ReLU Network Features. *arXiv* **2018**, arXiv:1803.08367.

36. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.
37. Han, Z.; Fu, Z.; Chen, S.; Yang, J. Contrastive embedding for generalized zero-shot learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021.
38. Verma, V.K.; Arora, G.; Mishra, A.; Rai, P. Generalized Zero-shot learning via synthesized examples. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 4281–4289.
39. Zhu, Y.; Xie, J.; Liu, B.; Elgammal, A. Learning feature-to-feature translator by alternating back-propagation for generative zero-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 9843–9853.
40. Atzmon, Y.; Chechik, G. Adaptive confidence smoothing for generalized zero-shot learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 11671–11680.
41. Huynh, D.; Elhamifar, E. Fine-grained generalized zero-shot learning via dense attribute-based attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–20 June 2020; pp. 4482–4492.
42. Min, S.; Yao, H.; Xie, H.; Wang, C.; Zha, Z.J.; Zhang, Y. Domain-aware Visual Bias Eliminating for Generalized Zero-Shot Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.
43. Zhongqi, Y.; Tan, W.; Hanwang, Z.; Qianru, S.; Xian-Sheng, H. Counterfactual Zero-Shot and Open-Set Visual Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–20 June 2021.
44. Ge, J.; Xie, H.; Min, S.; Zhang, Y. Semantic-guided reinforced region embedding for generalized zero-shot learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 1406–1414.
45. Chou, Y.Y.; Lin, H.T.; Liu, T.L. Adaptive and generative zero-shot learning. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
46. Liu, L.; Zhou, T.; Long, G.; Jiang, J.; Dong, X.; Zhang, C. Isometric propagation network for generalized zero-shot learning. *arXiv* **2021**, arXiv:2102.02038.
47. Zhu, Y.; Elhoseiny, M.; Liu, B.; Peng, X.; Elgammal, A. A generative adversarial approach for zero-shot learning from noisy texts. In Proceedings of the CVPR, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1004–1013.
48. Chen, S.; Xie, G.; Liu, Y.; Peng, Q.; Sun, B.; Li, H.; You, X.; Shao, L. Hsva: Hierarchical semantic-visual adaptation for zero-shot learning. In Proceedings of the Advances in Neural Information Processing Systems 34 (NeurIPS 2021), Virtual, 6–14 December 2021; Volume 34.