IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 19, 2022, accepted May 5, 2022, date of publication May 30, 2022, date of current version June 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3178800

Content-Attribute Disentanglement for Generalized Zero-Shot Learning

YOOJIN AN[1], SANGYEON KIM[2], YUXUAN LIANG[3], ROGER ZIMMERMANN[3], (Senior Member, IEEE), DONGHO KIM[4], AND JIHIE KIM[1]

[1]Department of Artificial Intelligence, Dongguk University, Seoul 04620, South Korea
[2]NAVER WEBTOON AI, Seongnam 13529, South Korea
[3]School of Computing, National University of Singapore, Singapore 119077
[4]Dongguk Institute of Convergence Education, Dongguk University, Seoul 04620, South Korea

Corresponding author: Jihie Kim (jihie.kim@dgu.edu)

This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the High-Potential Individuals Global Training Program) (2021-0-01549) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) (50%). This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2022-2020-0-01789) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) (50%).

ABSTRACT Humans can recognize or infer unseen classes of objects using descriptions explaining the characteristics (semantic information) of the classes. However, conventional deep learning models trained in a supervised manner cannot classify classes that were unseen during training. Hence, many studies have been conducted into generalized zero-shot learning (GZSL), which aims to produce system which can recognize both seen and unseen classes, by transferring learned knowledge from seen to unseen classes. Since seen and unseen classes share a common semantic space, extracting appropriate semantic information from images is essential for GZSL. In addition to semantic-related information (attributes), images also contain semantic-unrelated information (contents), which can degrade the classification performance of the model. Therefore, we propose a content-attribute disentanglement architecture which separates the content and attribute information of images. The proposed method is comprised of three major components: 1) a feature generation module for synthesizing unseen visual features; 2) a content-attribute disentanglement module for discriminating content and attribute codes from images; and 3) an attribute comparator module for measuring the compatibility between the attribute codes and the class prototypes which act as the ground truth. With extensive experiments, we show that our method achieves state-of-the-art and competitive results on four benchmark datasets in GZSL. Our method also outperforms the existing zero-shot learning methods in all of the datasets. Moreover, our method has the best accuracy as well in a zero-shot retrieval task. Our code is available at https://github.com/anyoojin1996/CA-GZSL.

INDEX TERMS Computer vision, deep learning, disentangled representation, generalized zero-shot learning.

I. INTRODUCTION

To classify images, humans can capture the characteristics—the semantic information—about objects, and use it to recognize a class of objects. Thanks to advances in deep learning technology, machines can mimic this ability, given large amounts of data, using supervised learning. Humans can recognize the class to which an object belongs using descriptions of the object from encyclopedias, even when they have never seen the class before. However, conventional supervised learning-based models cannot classify classes which are not seen during training. Therefore, if additional classes are added after training, the models must be re-trained from scratch.

To tackle this problem, several studies in a field known as zero-shot learning (ZSL), have been dedicated to enabling models to classify unseen classes [1]–[4]. The goal of conventional ZSL is to classify unseen classes using knowledge learned from seen classes. For generalized ZSL (GZSL), models are required to have the capacity to classify both seen

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyu Zhou.

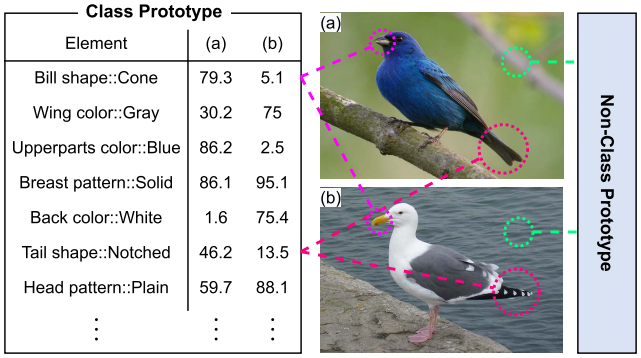


FIGURE 1. Example of class prototypes: two sets of ground truth class attributes share a common semantic space but are distinct with intensities. (a) Indigo bunting. (b) Western gull. The purple and the red lines indicate attributes associated with the class prototype of the birds, and the green lines indicate non-class attributes unrelated to the class prototype.

and unseen classes after training only with seen classes. Existing GZSL research [5]–[8] has branched into embedding-based and generative-based methods. Embedding-based methods aim to classify the unseen classes by mapping visual features into semantic vectors. Generative-based methods generate unseen visual features using the unseen semantic vectors and randomly initialized noise vectors. As recent generative-based methods, CE-GZSL [9] uses contrastive embedding to leverage instance-wise supervision. AGZSL [10] fuses adaptive and generative mechanisms, and supplements image-adaptive attention for GZSL. In this work, we focus on the generative-based method.

In the ZSL task, it is important to transfer knowledge learned from seen classes to unseen classes. Thereby, side information such as the descriptions mentioned earlier, is required to bridge the gap between seen and unseen classes. Researchers have utilized class prototypes [11]–[13], word embeddings [2], [14], and text descriptions [15], [16] as the side information. Recent work [6], [9], [17] has mainly focused on the class prototypes as side information. Class prototypes contain meaningful semantic knowledge describing the corresponding class. As shown in Fig. 1, the class prototypes represent the characteristics of classes. A set of elements of class prototypes is pre-defined, so seen and unseen classes share the same semantic space, with different intensities per class. A class prototype acts as a set of ground truth class attributes describing the characteristics of the class. Hence, it is crucial to correctly map visual features obtained from deep convolutional neural networks like ResNet-101 [18] into class prototypes. To align visual features into corresponding class prototypes, neural network models have to extract visual features similar to class prototypes, which correspond to class labels. However, as shown in Fig. 1, visual features also contain information which is not involved in class prototypes that can therefore degrade the performance of zero-shot classification.

To mitigate this problem, models need to disentangle the non-class and class attributes of images. We therefore define 1) features irrelevant to class prototypes as semantic-unrelated features, 2) features involved in class prototypes as semantic-related features, and 3) features extracted from ResNet-101 as visual features. As the prior state-of-the-art, SDGZSL disentangles the semantic-unrelated and semantic-related features using a single encoder. Then, it aligns the semantic-related features and class prototypes. However, we argue that semantic-unrelated and semantic-related features need to be extracted using independent encoders, because the feature spaces of the two groups are not identical. SDGZSL also uses a concatenation operator when reconstructing an original visual feature from two disentangled features. We also claim that aligning two features would produce better results than simple concatenation, as discussed in [19]. This contention is motivated by the style transfer research, which focuses on disentangling content and style representations. Many style transfer methods [20]–[22] have produced impressive improvements in performance, and have shown that the content-style disentanglement works. We can interpret styles as class attributes in the ZSL task. Therefore, we can disentangle semantic-unrelated information (*contents*) and semantic-related information (*attributes*) from visual features.

In this paper, we propose a novel content-attribute disentanglement architecture for generalized zero-shot learning (CA-GZSL). Our model encodes content-attribute codes from an original visual feature. The model learns content codes by calculating a reconstruction loss and attribute codes by measuring compatibility scores with class prototypes. When reconstruction, to fuse the content-attribute codes effectively, we use adaptive instance normalization (AdaIN) [19] which aligns the statistics of two different codes, leading to strong generalizability.

In summary, our main contributions are three-fold:

- We propose a novel content-attribute disentanglement architecture for generalized zero-shot learning (CA-GZSL). It comprises a visual feature generation module, a content-attribute disentanglement module, and an attribute comparator module.
- To the best of our knowledge, this is the first attempt to introduce adaptive instance normalization into the generative-based GZSL method to improve content-attribute disentanglement.
- The proposed method achieves state-of-the-art and competitive results on four datasets in GZSL, CZSL, and zero-shot retrieval tasks. Our approach is the first to obtain over 80% result on CUB in GZSL and over 50% result on aPY in CZSL.

II. RELATED WORK

A. GENERALIZED ZERO-SHOT LEARNING

The aim of zero-shot learning (ZSL) is to transfer knowledge from seen to unseen classes. The ZSL research can be divided into inductive and transductive approaches. In inductive ZSL

training [23]–[25], unseen class prototypes are used along with seen data. In transductive ZSL training [26]–[28], unlabeled visual features of unseen classes can be used in addition to unseen class prototypes and seen data. With respect to testing, ZSL is divided into conventional zero-shot learning (CZSL) and generalized zero-shot learning (GZSL). While CZSL predicts classes in unseen data, GZSL categorizes classes in both seen and unseen data. In general, GZSL is considered to be harder to achieve than CZSL, since models would be biased toward the seen data which is the only type used in training. Our method belongs to the inductive GZSL category.

GZSL can be achieved in two ways: embedding-based [2], [14], [29]–[32] or generative-based methods [8], [33]–[36]. The embedding-based method has focused on learning visual features and class prototypes by aligning them in a joint embedding space. As an embedding-based method, DAZLE [5] uses an attention mechanism to highlight important local features. DCEN [6] learns task-independent knowledge via contrastive learning, to transfer representations.

The generative-based methods have synthesized unseen visual features using generative adversarial networks (GANs) [37] or variational autoencoders (VAEs) [38], which transforms the ZSL problem into a supervised classification problem. CADA-VAE [7] leverages two VAEs to align the latent distributions of different modalities. TF-VAEGAN [35] uses a feedback module to reflect feedback from the decoder to the generator. CE-GZSL [9] is a hybrid framework which integrates generation and embedding-based methods using contrastive learning. Our approach belongs to generative-based GZSL.

B. CONTENT-STYLE DISENTANGLEMENT

Content-style disentanglement separates content and style representations from an image or two different images. This concept has been widely used in style transfer [19], image-to-image translation [39], and style classification [20] tasks. Numerous studies have shown enhanced qualitative results using content and style encoder-decoder architectures. They have used style information to stylize the content of an image. The content and attribute representations are then combined to reconstruct the original image, to demonstrate that they are complementary, and thus well-separated. As one of the operations combining content and style representations, adaptive instance normalization (AdaIN) [19] has been used, and has produced significant improvement. AdaIN is an instance normalization, which aligns feature statistics between two different feature distributions. It has also demonstrated a strong generalization ability.

MUNIT [39] provides an unsupervised image-to-image translation using content codes that are domain-invariant, and style codes that contain domain-specific properties. ALADIN [20] learns fine-grained style similarities among digital artworks by leveraging content-style disentanglement. ALADIN shows the remarkable impact of style representations constructed using a set of style codes. In GZSL, some approaches use a disentangling framework. DLFZRL [40] incorporates a hierarchical disentanglement structure for the discrimination of latent features. SDGZSL [41] uses a total correlation penalty for the disentanglement of semantic-related and semantic-unrelated features.

Unlike the existing works, we define the style of an image as a set of attributes. Thereby, we focus on the impact of the content-attribute disentanglement architecture using an encoder-decoder network equipped with AdaIN to improve the attribute representation.

III. APPROACH

A. PROBLEM DEFINITION

For zero-shot learning (ZSL), we use a seen dataset $\mathcal{S}$ and an unseen dataset $\mathcal{U}$. d_{res} is the dimensionality of a visual feature extracted using ResNet-101, and d_{att} is the dimensionality of a set of class prototypes. The seen dataset $\mathcal{S}$ is defined as $\mathcal{S} = \{x_s, y_s, a_{y_s} | x_s \in \mathcal{X}^s, y_s \in \mathcal{Y}^s, a_{y_s} \in \mathcal{A}^s\}$, where $x_s \in \mathbb{R}^{d_{res}}$ is a d_{res}-dimensional visual feature extracted from ResNet-101, $y_s \in \mathbb{R}^1$ is a label in the seen classes, and $a_{y_s} \in \mathbb{R}^{d_{att}}$ is a d_{att}-dimensional class prototype of the class y_s. The unseen dataset $\mathcal{U}$ is defined as $\mathcal{U} = \{x_u, y_u, a_{y_u} | x_u \in \mathcal{X}^u, y_u \in \mathcal{Y}^u, a_{y_u} \in \mathcal{A}^u\}$, where $x_u \in \mathbb{R}^{d_{res}}$ is a d_{res}-dimensional visual feature extracted from ResNet-101, $y_u \in \mathbb{R}^1$ is a label of the unseen classes, and $a_{y_u} \in \mathbb{R}^{d_{att}}$ is a d_{att}-dimensional class prototype of the class y_u. The two classes—seen classes and unseen classes—are disjoint, $\mathcal{Y}^s \cap \mathcal{Y}^u = \emptyset$.

B. MODEL OVERVIEW

Our method is divided into two stages: the first stage is introduced in Fig. 2, and corresponds to Subsections 1) to 4), and the second stage is for final classification, corresponding to Subsection 5).

In Fig. 2, the proposed architecture comprises three modules: (a) a visual feature generation module consisting of a variational encoder Q and a variational decoder P; (b) a content-attribute disentanglement module consisting of a content encoder E, an attribute encoder H, an adaptive instance normalization (AdaIN), and a decoder D; (c) an attribute comparator module consisting of a comparator T. First, the visual feature generation module synthesizes unseen visual features from unseen class prototypes using a variational autoencoder (VAE). To let the VAE know how to correctly synthesize visual features, we first train it with seen visual features. Then, the content-attribute disentanglement module encodes the content and attribute codes using encoders. We combine the content and attribute codes using AdaIN [19] and reconstruct the original visual features from the combined codes using the decoder. Finally, the attribute comparator module measures the compatibility scores between the attribute codes and the class prototypes, and makes the attribute codes resemble the corresponding class prototypes.

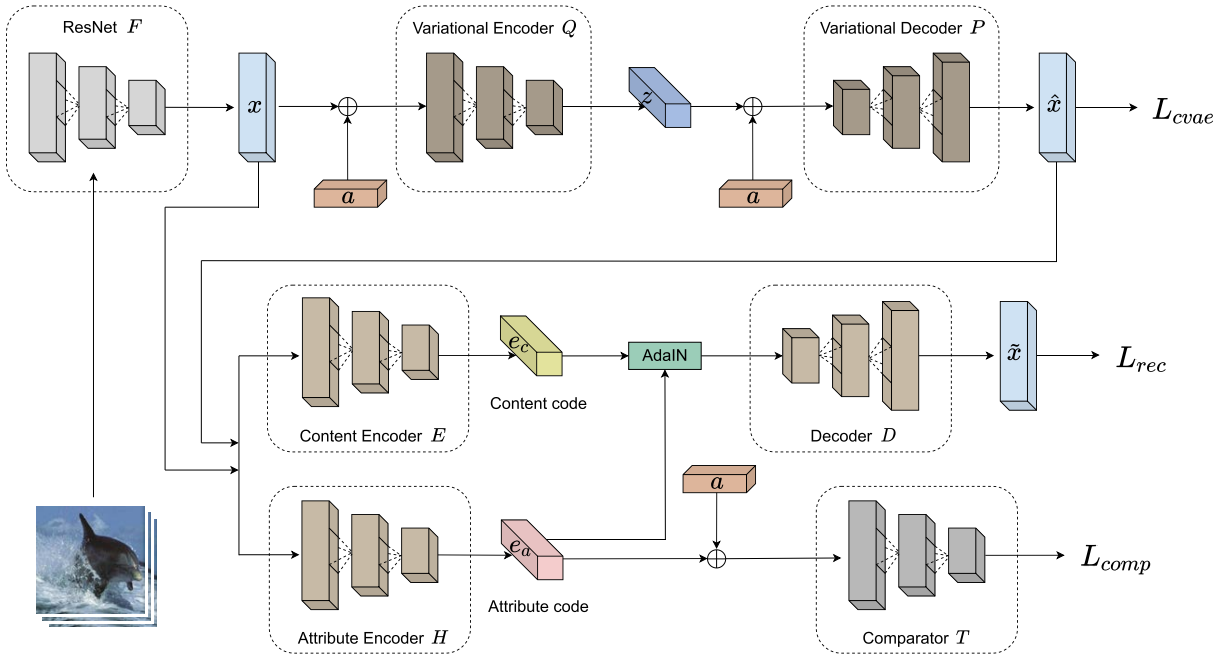


FIGURE 2. Illustration of content-attribute disentanglement architecture.

1) VISUAL FEATURE GENERATION MODULE

There are many generative-based GZSL approaches which use a VAE to synthesize unseen visual features [7], [8], [33], [35], as unseen visual features are not allowed to be used in training. We use a conditional variational autoencoder (CVAE) to generate synthesized visual features $\hat{x} \in \mathbb{R}^{d_{res}}$ conditioned on seen or unseen class prototypes. The CVAE first generates synthesized seen visual features $\hat{x}_s \in \mathbb{R}^{d_{res}}$ using the seen dataset $\mathcal{S}$ for use in training. The objective function of the CVAE is formulated as:

$$L_{cvae} = \mathbb{E}_{q(z|x_s,a_s)}[\log p(x_s|z,a_s)] - D_{KL}[q(z|x_s,a_s)\,\|\,p(z|a_s)] \tag{1}$$

where the first term is the reconstruction loss and the second term is the Kullback-Leibler (KL) divergence between $q(z|x_s,a_s)$ and $p(z|a_s)$. $x_s \in \mathbb{R}^{d_{res}}$ and $a_s \in \mathbb{R}^{d_{att}}$ are the seen visual features and the seen class prototypes. The CVAE encoder Q models $q(z|x_s,a_s)$ to produce the latent variables z using seen visual features x_s and seen class prototypes a_s. The CVAE decoder P models $p(z|a_s)$ and $p(x_s|z,a_s)$ to synthesize visual features from the latent variables z and the seen class prototypes a_s. We use seen visual features x_s and synthesized seen visual features $\tilde{x}_s$ as an input to the networks in the content-attribute disentanglement module.

As previously mentioned, it is noteworthy that a_s and a_u share a common semantic space. Thus, after training the CVAE, the CVAE decoder can synthesize unseen visual features $\hat{x}_u \in \mathbb{R}^{d_{res}}$ using unseen class prototypes $a_u \in \mathbb{R}^{d_{att}}$. Both seen visual features and synthesized unseen visual features are used in training a zero-shot classifier. We use seen visual features extracted by ResNet-101, which was pre-trained on ImageNet [42] or fine-tuned with seen class images.

2) CONTENT-ATTRIBUTE DISENTANGLEMENT MODULE

We define semantic-unrelated information as *contents* and semantic-related information as *attributes* that represent the styles of classes. The content-attribute disentanglement module comprises four main parts: a content encoder, an attribute encoder, an AdaIN, and a decoder. The content-attribute disentanglement module divides visual features into the contents and the attributes. The content encoder $E : \mathbb{R}^{d_{res}} \to \mathbb{R}^{d_{att}}$ and the attribute encoder $H : \mathbb{R}^{d_{res}} \to \mathbb{R}^{d_{att}}$ map a visual feature x into a content code e_c and an attribute code e_a, respectively. Then, the content code and attribute code can be denoted as:

$$e_c = E(x) \tag{2}$$

$$e_a = H(x) \tag{3}$$

where $e_c, e_a \in \mathbb{R}^{d_{att}}$ are d_{att}-dimensional representations.

We use AdaIN which aligns two different feature statistics to reconstruct an original visual feature. AdaIN takes as input a content code e_c and an attribute code e_a, and aligns the channel-wise mean and standard deviation of e_c to match those of e_a. AdaIN is defined as follows:

$$AdaIN(e_c, e_a) = \sigma(e_a)\left(\frac{e_c - \mu(e_c)}{\sigma(e_c)}\right) + \mu(e_a) \tag{4}$$

where $\sigma(e_c)$ and $\mu(e_c)$ are the standard deviation and mean, respectively, of a content code, respectively. $\sigma(e_a)$ and $\mu(e_a)$ are the standard deviation and mean, respectively, of an attribute code. We first normalize the content code e_c with

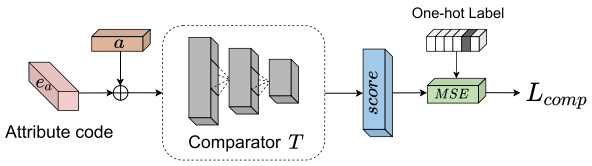


FIGURE 3. Illustration of attribute comparator module.

the mean and standard deviation of the content code $\mu(e_c)$ and $\sigma(e_c)$, scale the normalized content code with standard deviation of the attribute code $\sigma(e_a)$, and shift it with the mean of the attribute code $\mu(e_a)$.

The reconstructed visual features obtained from the content and attribute codes should resemble the original visual features. Therefore, we measure the reconstruction loss to learn the content codes from the original and reconstructed visual features. The decoder $D : \mathbb{R}^{d_{att}+d_{att}} \rightarrow \mathbb{R}^{d_{res}}$ reconstructs the original visual feature from the aligned codes $AdaIN(e_c, e_a)$. The reconstructed visual feature $\tilde{x}$ and the reconstruction loss function can be formulated as:

$$\tilde{x} = D(AdaIN(e_c, e_a)) \tag{5}$$

$$L_{rec} = \sum_{x \in \mathcal{X}^s} \| x - \tilde{x} \|^2 \tag{6}$$

where x is an original visual feature. $\tilde{x}$ is a reconstructed visual feature from AdaIN followed by the decoder D. We measure the mean squared error (MSE) between the visual seen feature x and the reconstructed visual features $\tilde{x}$ as the reconstruction loss.

3) ATTRIBUTE COMPARATOR MODULE

Inspired by [43], we adopt an attribute comparator module to let models learn the compatibilities between attribute codes and class prototypes. As shown in Fig. 3, we concatenate an attribute code e_a and a class prototype a, as an input to the comparator T. The comparator T measures the compatibility score between e_a and a, while learning to maximize the score. The compatibility loss function can be formulated as:

$$L_{comp} = \sum_{i=1}^{N^s} \| T(e_a, a_i) - \varphi(y_i) \|^2 \tag{7}$$

where $y \in \mathcal{Y}^s$ indicates a ground-truth label and $\varphi(y)$ indicates a one-hot label of y. We calculate the MSE between the compatibility score $T(e_a, a_i)$ and the one-hot label $\varphi(y)$ to measure the compatibility loss.

4) TOTAL LOSS

Consequently, the total loss can be formulated as:

$$L_{total} = \lambda_1 L_{cvae} + \lambda_2 L_{rec} + \lambda_3 L_{comp} \tag{8}$$

where λ_1, λ_2, and λ_3 are the factors contributing to each loss, controlling each impact. We use the Optuna package [44] to search for the hyperparameters λ_1, λ_2, and λ_3.

TABLE 1. Dataset statistics of CUB, AWA2, FLO and aPY. Seen and Unseen is the number of seen and unseen classes, respectively. * indicates that FLO has 1024-d embeddings extracted from a network instead of manual attributes.

Dataset	Seen	Unseen	Attribute	Total
CUB	100 + 50 (150)	50	312	11,788
AWA2	27 + 13 (40)	10	85	37,322
FLO	62 + 20 (82)	20	*1,024	8,189
aPY	15 + 5 (20)	12	64	15,339

5) GENERALIZED ZERO-SHOT CLASSIFICATION

As unseen visual features are not used in training, we generate unseen visual features for generalized zero-shot classification. The decoder of CVAE generates unseen visual features with an unseen class prototype a_u and a Gaussian noise z. The attribute encoder H encodes attribute codes from the synthesized unseen visual features. Then, we concatenate the seen attribute codes and the synthesized unseen attribute codes. Using these codes, we train a classifier. For the classifier, we use only one linear layer, to make the system consistent with existing work. Because we have a complex architecture with which to effectively extract attributes in the first stage, the classifier in the second stage has a simple architecture. This classifier can be formulated as:

$$\hat{y} = \underset{\tilde{y} \in \mathcal{Y}^s \cup \mathcal{Y}^u}{\arg\max}\, \text{softmax}(l(e_a)) \tag{9}$$

where l is a linear layer followed by a softmax. $e_a \in \mathbb{R}^{d_{att}}$ is an attribute code extracted from the attribute encoder H.

IV. EXPERIMENTS

A. DATASETS

We use four popular benchmark datasets, Caltech-UCSD Birds-200-2011 (CUB) [49], Animals with Attributes 2 (AWA2) [50], Oxford Flowers (FLO) [51] and a Pascala Yahoo (aPY) [12] to measure the CZSL and GZSL performance. For evaluating the GZSL performance, we split each dataset into training-seen, test-seen and test-unseen classes as following proposed split which is suggested in [50].

CUB dataset is a fine-grained bird dataset that contains 11,788 CUB images, including 7,057 training-seen images, 1,764 test-seen images, and 2,967 test-unseen images. The total number of classes is 200, divided into 150 seen classes and 50 unseen classes. CUB has 312 attributes.

AWA2 dataset is a coarse-grained animal dataset. The total number of AWA2 images is 37,322, and this is composed of 23,527 training-seen images, 5,882 test-seen images, and 7,913 test-unseen images. The total number of classes is 50, divided into 40 seen classes and 10 unseen classes. AWA2 has 85 attributes.

FLO dataset is a fine-grained flower dataset. The total number of FLO images is 8,189, and this is composed of 1,640 training-seen images, 5,394 test-seen images, and 1,155 test-unseen images. The total number of classes is 102, divided into 82 seen classes and 20 unseen classes.

TABLE 2. Results of the GZSL methods. There are three blocks in the table. The first block concerns embedding-based methods, the second block concerns generative-based methods, and the last block concerns our method. U is $acc_{\mathcal{Y}^u}$ and S is $acc_{\mathcal{Y}^s}$ for simplicity. The best H results are highlighted in bold, since H is the major metric in GZSL. * indicates fine-tuned dataset, which was fine-tuned using seen class images.

Method	CUB			AWA2			FLO			aPY		
	GZSL			GZSL			GZSL			GZSL		
	U	S	H	U	S	H	U	S	H	U	S	H
RN [43]	38.1	61.1	47.0	30.0	93.4	45.3	-	-	-	-	-	-
TCN [45]	52.6	52.0	52.3	61.2	65.8	63.4	-	-	-	-	-	-
DAZLE [5]	56.7	59.6	58.1	60.3	75.7	67.1	-	-	-	-	-	-
APN [46]	65.3	69.3	67.2	56.5	78.0	65.5	-	-	-	-	-	-
DCEN [6]	63.8	78.4	70.4	62.4	81.7	70.8	-	-	-	37.5	61.6	46.7
GEM-ZSL [17]	64.8	77.1	70.4	64.8	77.5	70.6	-	-	-	-	-	-
f-VAEGAN-D2 [33]	48.4	60.1	53.6	57.6	70.6	63.5	56.8	74.9	64.6	-	-	-
f-VAEGAN-D2* [33]	63.2	75.6	68.9	57.1	76.1	65.2	63.3	92.4	75.1	-	-	-
CADA-VAE [7]	51.6	53.5	52.4	55.8	75.0	63.9	51.6	75.6	61.3	31.7	55.1	40.3
DLFZRL [40]	-	-	51.9	-	-	60.9	-	-	-	-	-	38.5
OCD-CVAE [8]	44.8	59.9	51.3	59.5	73.4	65.7	-	-	-	-	-	-
IZF [47]	52.7	68.0	59.4	60.6	77.5	68.0	-	-	-	42.3	60.5	49.8
TF-VAEGAN [35]	52.8	64.7	58.1	59.8	75.1	66.6	62.5	84.1	71.7	-	-	-
TF-VAEGAN* [35]	63.8	79.3	70.7	55.5	83.6	66.7	69.5	92.5	79.4	-	-	-
E-PGN [48]	52.0	61.1	56.2	52.6	83.5	64.6	71.5	82.2	76.5	-	-	-
CE-GZSL [9]	63.9	66.8	65.3	63.1	78.6	70.0	69.0	78.7	73.5	-	-	-
AGZSL [10]	41.4	49.7	45.2	65.1	78.9	71.3	-	-	-	35.1	65.5	45.7
AGZSL* [10]	69.2	76.4	72.6	69.0	86.5	**76.8**	-	-	-	36.2	58.6	44.8
SDGZSL [41]	59.9	66.4	63.0	64.6	73.6	68.8	83.3	90.2	86.6	38.0	57.4	45.7
SDGZSL* [41]	73.0	77.5	75.1	69.6	78.2	73.7	86.1	89.1	87.8	39.1	60.7	47.5
CA-GZSL (Ours)	65.9	63.7	64.8	65.3	74.0	69.4	83.9	90.8	87.2	38.3	52.4	44.3
CA-GZSL* (Ours)	74.6	80.0	**77.2**	68.8	82.4	75.0	85.4	94.5	**89.7**	42.5	62.0	**50.5**

FLO has 1024-dimensional attribute embeddings extracted from a character-based CNN-RNN using fine-grained visual descriptions [15].

aPY dataset is a coarse-grained dataset. The total number of aPY images is 15,339, and this is composed of 5,932 training-seen images, 1,483 test-seen images, and 7,924 test-unseen images. The total number of classes is 32, divided into 20 seen classes and 12 unseen classes. aPY has 64 attributes.

We also use fine-tuned datasets from [33], where ResNet-101 is fine-tuned by the seen class images of each dataset.

B. IMPLEMENTATION DETAILS

We use three fully-connected layers with 2048 hidden units for the VAE encoder Q and VAE decoder P. Leaky ReLU is used as an activation function. We use the same hyperparameters of the visual feature generation module as SDGZSL for fair comparisons. We use the 64 mini-batch sizes in all the datasets. For the content and attribute encoders E, H and the decoder D, we use two fully-connected layers with d_{att} hidden units. We optimize all the networks with the Adam optimizer for each module. Two fully-connected layers for the comparator module T are used with 2048 hidden units. We use a single fully-connected layer for the classifier for both CZSL and GZSL. To search hyperparameters, we use the Optuna package [44]. We made the source code available for the detailed hyperparameters. All the models are implemented with the PyTorch framework v1.7.0 [52]. We use a single RTX 2080 Ti 11GB GPU for each training.

C. EVALUATION METRICS

We assess the performance of conventional zero-shot learning (CZSL) and generalized zero-shot learning (GZSL) with average per-class top-1 accuracy ($T1$) and harmonic mean (H) between seen and unseen $T1$ accuracies as presented in [50]. $T1$ is defined as follows to evaluate ZSL:

$$acc_{\mathcal{Y}} = \frac{1}{\|\mathcal{Y}\|} \sum_{c=1}^{\|\mathcal{Y}\|} \frac{\text{Number of correct predictions in c}}{\text{Total number of samples in c}} \tag{10}$$

where $\|\mathcal{Y}\|$ means the number of classes in $\mathcal{Y}$. The equation of H is defined as follows to evaluate GZSL:

$$H = \frac{2 \times acc_{\mathcal{Y}^s} \times acc_{\mathcal{Y}^u}}{acc_{\mathcal{Y}^s} + acc_{\mathcal{Y}^u}} \tag{11}$$

where $acc_{\mathcal{Y}^s}$ and $acc_{\mathcal{Y}^u}$ mean the average per-class top-1 accuracies for seen and unseen classes, respectively.

D. COMPARISON WITH STATE-OF-THE-ARTS

Previous generative-based methods have used visual features extracted from ResNet-101 pre-trained on ImageNet or fine-tuned on the seen class images of each dataset. We measure the CZSL and GZSL performance of our CA-GZSL method.

TABLE 3. Results of the CZSL methods. The upper part shows embedding-based methods, the second block shows generative-based methods. The last block shows our method. The best results are highlighted in bold. * indicates the fine-tuned dataset, which was fine-tuned using seen class images.

Method	CUB	AWA2	FLO	aPY
RN [43]	55.6	64.2	-	-
TCN [45]	59.5	71.2	-	-
DAZLE [5]	65.9	-	-	-
APN [46]	72.0	68.4	-	-
GEM-ZSL [17]	77.8	67.3	-	-
f-VAEGAN-D2 [33]	61.0	71.1	67.7	-
f-VAEGAN-D2* [33]	72.9	70.3	70.4	-
CADA-VAE [7]	60.4	64.0	65.2	-
DLFZRL [40]	61.8	70.3	-	46.7
OCD-CVAE [8]	60.3	71.3	-	-
IZF [47]	67.1	74.5	-	44.9
TF-VAEGAN [35]	64.9	72.2	70.8	-
TF-VAEGAN* [35]	74.3	73.4	74.7	-
E-PGN [48]	72.4	73.4	85.7	-
CE-GZSL [9]	77.5	70.4	70.6	-
AGZSL [10]	57.2	73.8	-	41.0
AGZSL* [10]	77.2	76.4	-	43.7
SDGZSL [41]	75.5	72.1	85.4	45.4
SDGZSL* [41]	78.5	74.3	86.9	47.0
CA-GZSL (Ours)	77.5	74.8	88.7	44.7
CA-GZSL* (Ours)	**81.3**	**79.0**	**88.1**	**47.9**

We select the recent state-of-the-art embedding-based and generative-based methods, as listed in Table 2 and 3.

1) BASELINES

We compare CA-GZSL with the recent CZSL and GZSL methods, composed of embedding-based methods such as RN [43], TCN [45], DAZLE [5], APN [46], DCEN [6] and GEM-ZSL [17], and generative-based methods such as f-VAEGAN-D2 [33], CADA-VAE [7], DLFZRL [40], OCD-CVAE [8], IZF [47], TF-VAEGAN [35], E-PGN [48], CE-GZSL [9], AGZSL [10] and SDGZSL [41].

2) RESULTS OF GENERALIZED ZERO-SHOT LEARNING

In Table 2, we show the results of the evaluation of the GZSL performance and compare our CA-GZSL with recent GZSL methods. Our method surpasses the baselines by about 2% on CUB, FLO, and aPY, and obtains the second-best H result on AWA2. Notably, on the aPY dataset, our method is the first to achieve performance over 50 on H compared with the existing methods. Our intuition for GZSL is in accord with that of SDGZSL, the previous best model among the generative-based methods. Both aim at disentangling semantic-related and semantic-unrelated information from visual features. To do so, SDGZSL applies a total correlation penalty to ensure independence between semantic-related and semantic-unrelated features by dividing a feature generated from a single encoder. SDGZSL combines the two features to reconstruct the original images by concatenating the two features. However, we assume that it is difficult for a single encoder to separate two independent features. For better disentanglement, we introduce two different encoders to encode content and attribute codes from visual features. We use AdaIN when combining content and attribute codes, to improve the generalization ability, and then we reconstruct the original images using the combined codes. Compared with the performance of SDGZSL, as shown in Table 2, for all the fine-tuned datasets, our method outperforms SDGZSL in H results. This result indicates that our approach is more effective in learning discriminative attribute features by effectively disentangling the contents and attributes from the visual features.

3) RESULTS OF CONVENTIONAL ZERO-SHOT LEARNING

We report the CZSL performance in Table 3. Our method achieves state-of-the-art results on all of the datasets. Notably, on CUB, we are the first work to obtain a performance over 80%. Table 3 shows that we outperform SDGZSL in all fine-tuned datasets. In particular, on AWA2, our model outperforms SDGZSL by about 4.7%. The results listed in Table 2 and 3 indicate that our method is more generalizable than the alternatives in ZSL tasks.

E. ZERO-SHOT RETRIEVAL RESULTS

1) ZERO-SHOT RETRIEVAL PROTOCOL

We follow the zero-shot retrieval protocol proposed in SDGZSL [41]. First, ResNet-101 extracts the visual features from all unseen images. Then, the attribute encoder encodes the unseen visual features into attribute codes. The attribute codes act as reference features. Third, the VAE synthesizes N unseen visual features per class. The attribute encoder encodes the unseen visual features into synthesized unseen attribute codes. The total number of synthesized unseen attribute codes is $N \times \|\mathcal{Y}^u|$, where $\|\mathcal{Y}^u\|$ is the number of unseen classes. Fourth, we average N synthesized unseen attribute codes to produce a representative feature of each class. The $\|\mathcal{Y}^u\|$ representative features act as query features. Lastly, we measure the cosine similarity between a query feature and the reference features, and rank the reference features by similarity in descending order.

2) QUANTITATIVE EVALUATION

Fig. 4 shows a comparison of the zero-shot retrieval performance of CVAE, SDGZSL, and CA-GZSL (ours). The metric we use to evaluate the zero-shot retrieval performance is the mean average precision at k (mAP@k). Our method, CA-GZSL, outperforms the other approaches in all of the datasets except for mAP@25 on AWA2. For CUB and aPY, CA-GZSL has notably better performance of mAP than the others. Specifically, on CUB, CA-GZSL outperforms SDGZSL by 14.5%, 23.3%, and 25.2% in mAP@100, 50, and 25, respectively. On aPY, CA-GZSL surpasses SDGZSL by 1.9%, 2.6%, and 5.5% in mAP@100, 50, and 25, respectively. On FLO, CA-GZSL shows better performance than

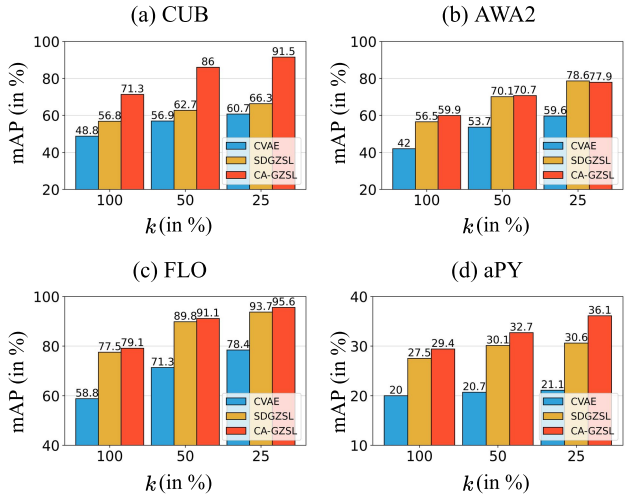


FIGURE 4. Comparison of the performance of zero-shot retrieval on unseen classes.

SDGZSL by 1.6%, 1.3%, and 1.9% in mAP@100, 50, and 25. On AWA2, CA-GZSL is better than SDGZSL by 3.4% and 0.6% in mAP@100 and 50, respectively. In contrast, SDGZSL is slightly higher than CA-GZSL, by 0.7% in mAP@25. Since our method outperforms SDGZSL in almost all of the evaluations, we argue that the attribute codes extracted using our method produce better discriminative information than SDGZSL, which helps to distinguish unseen classes, resulting in performance improvement.

3) QUALITATIVE EVALUATION

Fig. 5 illustrates the unseen images retrieved using CA-GZSL on AWA2. We perform the zero-shot retrieval according to the protocol described in Subsection E.1 of the experiments Section. As shown in Fig. 5, our model tends to be confused between blue whale and dolphin images, although the first false image in the blue whale column was mislabeled as a dolphin. Because the correct label is the blue whale, it can be seen that the model answered correctly. The model is more likely to be confused between walrus and seal images. At first glance, both classes look very similar. Also, the model is more likely to be confused between giraffe and bobcat images. Both classes have spot patterns, so the model tends to identify the discriminative patterns. Notably, the classes of the top three false predictions are consistent. We argue that our model can consistently identify meaningful attribute information from various images.

F. MODEL ANALYSIS

1) ABLATION STUDY

In Table 4, we show the results of an ablation study of all four datasets, which is used to measure the impact of each component of our CA-GZSL. We first evaluate the performance of the visual feature generation module, as a baseline. Then, we evaluate the performance of the content-disentanglement

FIGURE 5. Images from zero-shot retrieval on unseen classes. The highest-ranked image is at the top of each true and false prediction column.

TABLE 4. Ablation study of our CA-GZSL on four benchmarks. T1 is the top-1 accuracy and H is the harmonic mean. The best results are highlighted in bold.

Methods	CUB		AWA2		FLO		aPY	
	T1	**H**	**T1**	**H**	**T1**	**H**	**T1**	**H**
Baseline	59.2	51.7	71.5	67.1	68.5	66.4	40.2	40.6
+ L_{rec}	81.1	76.9	75.0	72.8	87.7	89.0	42.3	45.0
+ L_{comp}	**81.3**	**77.2**	**79.0**	**75.0**	**88.1**	**89.7**	**47.8**	**50.5**

module and the attribute comparator module, by gradually adding the reconstruction loss L_{rec} and the comparator loss L_{comp}. When L_{rec} is added to the baseline, the GZSL performance is enhanced by 25.2% (CUB), 5.7% (AWA2), 22.6% (FLO), and 4.4% (aPY), and the CZSL performance is enhanced by 21.9% (CUB), 3.5% (AWA2), 19.2% (FLO), and 2.1% (aPY). When L_{comp} is added, there are GZSL performance improvements of 0.3% (CUB), 2.2% (AWA2), 0.7% (FLO), and 5.5% (aPY), and CZSL performance improvements of 0.2% (CUB), 4.0% (AWA2), 0.4% (FLO), and 5.5% (aPY). In both ZSL performance ($T1$) and GZSL performance (H), the final model obtains the best results.

2) IMPACT OF THE NUMBER OF SYNTHESIZED FEATURES

The generative-based GZSL works synthesize unseen visual features, as only seen visual features are available in training. Thus, in this experiment, we evaluate the impact of the number of synthesized visual features. Fig. 6 shows the seen accuracy (S), unseen accuracies (U), and harmonic mean (H) when changing the number of synthesized visual features from 5 to 4,000. When the lowest number (5) of synthesized

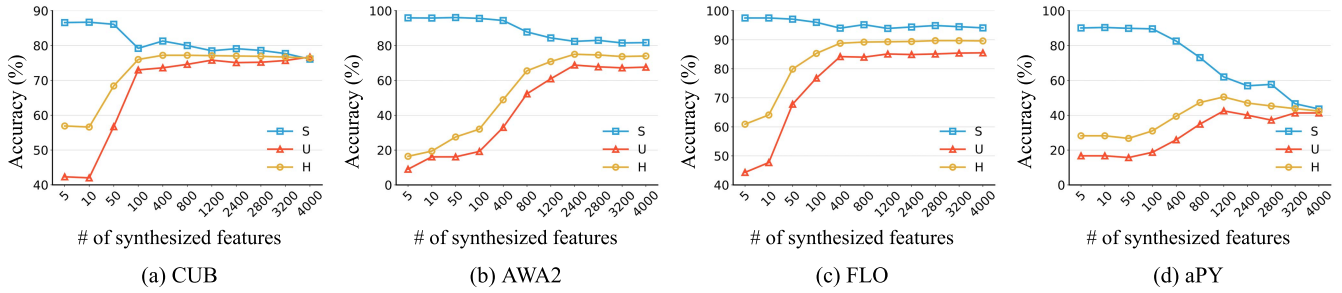


FIGURE 6. Results of GZSL performance with different numbers of synthesized unseen visual features.

TABLE 5. Results of GZSL performance with different feature fusion operators which combine the content-attribute codes. 'Concat' and 'Sum' mean the concatenation and the element-wise summation, respectively.

Operator	CUB				AWA2				FLO				aPY			
	T1	**U**	**S**	**H**	**T1**	**U**	**S**	**H**	**T1**	**U**	**S**	**H**	**T1**	**U**	**S**	**H**
Concat	78.8	74.0	80.2	77.0	76.4	63.0	79.9	70.4	86.0	83.6	93.2	88.1	46.5	34.5	65.6	45.2
Sum	80.0	73.7	80.8	77.1	77.2	62.8	81.8	71.1	86.0	82.7	92.4	87.3	45.3	35.5	63.3	45.5
AdaIN	**81.3**	**74.6**	80.0	**77.2**	**79.0**	**68.8**	82.4	**75.0**	**88.1**	**85.4**	94.5	**89.7**	**47.9**	**42.5**	62.0	**50.5**

visual features is used, S is the highest, and U is the lowest in all datasets. There is a trade-off between S and U. When S goes up, U goes down, and vice versa. On AWA2, our model achieves the best H result (75%) in 2,400. On CUB, we observe the best H performance (77.2%) in 400 and 800. On FLO, it produces the best H result (89.7%) in both 2,800 and 3,200. On aPY, it shows the best H performance (50.5%) in 1,200.

3) IMPACT OF FEATURE FUSION OPERATORS

We evaluate the impact of several fusion operators which combine content and attribute codes. As listed in Table 5, the 'Concat' means the concatenation, and the 'Sum' means the element-wise summation. In this experiment, AdaIN outperforms others on all four datasets. Notably, on AWA2, AdaIN obtains 4.6% and 3.9% higher GZSL performances than 'Concat' and 'Sum,' respectively. On aPY, AdaIN is better by 5.3% and 5% than 'Concat' and 'Sum,' respectively.

4) T-SNE VISUALIZATION

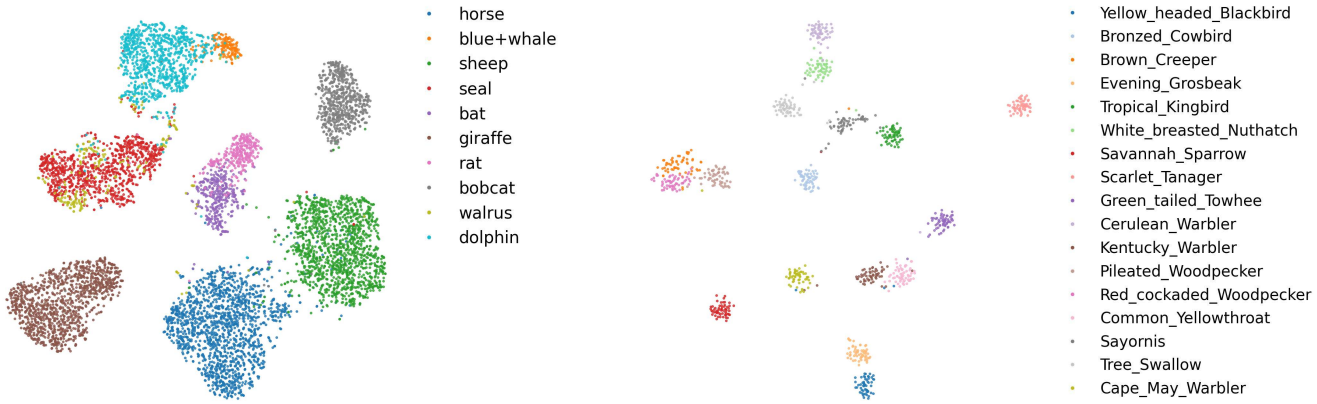
We visualize the attribute codes using the t-distributed stochastic neighbor embedding (t-SNE) algorithm [53], as shown in Fig. 7. The attribute encoder in our architecture encodes the attribute codes from unseen visual features. As illustrated in Fig. 7 (a), most of the clusters are isolated from each other on AWA2. In contrast, the seal and the walrus clusters are most closely mingled, and are hard to distinguish. The blue whale and dolphin clusters are close to each other. The bat and rat clusters are also close to each other. As shown in Fig. 5, these pairs of classes share similar attributes, leading to difficulties in discrimination. As shown in Fig. 7 (b), we randomly select 20 classes out of a total of 50 classes for clear visualization on CUB. Our method produces discriminative attribute codes, even though CUB has fine-grained classes that require models to catch more subtle differences in attributes than coarse-grained classes. Given this ability, our method outperforms the others by a significant margin in the zero-shot retrieval task, as shown in Fig. 4.

5) ATTRIBUTE SIMILARITY ANALYSIS

We conduct attribute similarity analysis using attribute codes extracted from an attribute encoder and images. We first analyze attribute similarities between unseen and seen classes. As shown in Fig. 8, the blue whale that belongs to an unseen class is similar to the humpback whale that belongs to a seen class. Likewise, the rat is similar to the mouse and the hamster, and the walrus is similar to the hippopotamus. In attribute similarities among unseen classes, there is a high degree of similarity among classes that are similar in appearance, as shown in Fig. 5. This analysis shows the attribute encoder can produce consistent and discriminative attribute codes from seen and unseen images.

V. DISCUSSION

We investigate the effectiveness of content-attribute disentanglement architecture for generalized zero-shot learning (GZSL). Our method for separating contents and attributes from images is found to outperform most of the existing approaches in GZSL and ZSL, following extensive experiments. As mentioned in Subsection E.2 of the experiments Section, the image of the first false prediction in the blue whale column was mislabeled as a dolphin, but the blue whale is the correct answer. We believe that refining the datasets widely used in GZSL would be valuable as future work. Although our method achieves state-of-the-art performance on most datasets, there is a limitation in classifying classes that have similar class prototypes, as shown in Fig. 5.



(a) AWA2 (b) CUB

FIGURE 7. t-SNE visualization of unseen attribute codes.

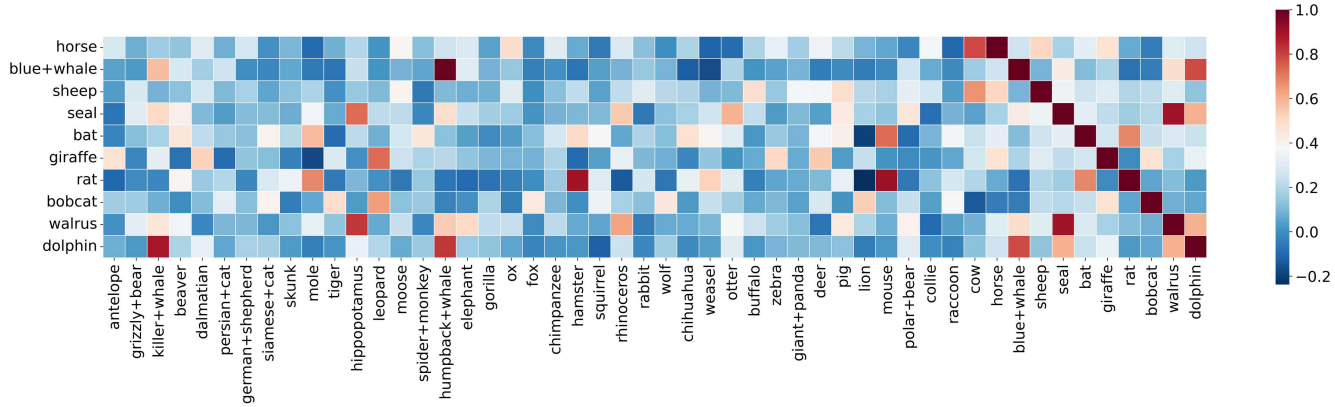


FIGURE 8. Illustration of similarities among attribute codes on AWA2.

We argue that it is hard to tackle the problem of subtle differences in values of class attributes only using the class prototype information. Therefore, auxiliary information such as knowledge graphs or additional text descriptions would be helpful to improve the zero-shot inference abilities of models.

VI. CONCLUSION

In this paper, we propose a novel content-attribute disentanglement architecture for generalized zero-shot learning, consisting of a visual feature generation module, a content-attribute disentanglement module, and an attribute comparator module. In addition, we present the first attempt to utilize adaptive instance normalization to improve disentanglement and generalization ability. Our method recognizes discriminative class attributes from visual features in both zero-shot classification and retrieval tasks, as demonstrated by extensive experiments. We evaluate our approach on four benchmark datasets widely used in ZSL. Compared to existing GZSL approaches, our method achieves state-of-the-art results using the CUB, FLO, and aPY datasets, and competitive results on AWA2. We will utilize additional information, such as knowledge graphs, in our architecture to discriminate classes with similar attributes in future work.

ACKNOWLEDGMENT

(Yoojin An and Sangyeon Kim are co-first authors.)

REFERENCES

[1] S. Min, H. Yao, H. Xie, Z.-J. Zha, and Y. Zhang, "Domain-oriented semantic embedding for zero-shot learning," *IEEE Trans. Multimedia*, vol. 23, pp. 3919–3930, 2020.

[2] A. Frome, G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 2121–2129.

[3] G.-S. Xie, L. Liu, F. Zhu, F. Zhao, Z. Zhang, Y. Yao, J. Qin, and L. Shao, "Region graph embedding network for zero-shot learning," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 562–580.

[4] S. Min, H. Yao, H. Xie, Z.-J. Zha, and Y. Zhang, "Domain-specific embedding network for zero-shot recognition," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 2070–2078.

[5] D. Huynh and E. Elhamifar, "Fine-grained generalized zero-shot learning via dense attribute-based attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4483–4493.

[6] C. Wang, X. Chen, S. Min, X. Sun, and H. Li, "Task-independent knowledge makes for transferable representations for generalized zero-shot learning," 2021, *arXiv:2104.01832*.
[7] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata, "Generalized zero- and few-shot learning via aligned variational autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8247–8255.
[8] R. Keshari, R. Singh, and M. Vatsa, "Generalized zero-shot learning via over-complete distribution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13300–13308.
[9] Z. Han, Z. Fu, S. Chen, and J. Yang, "Contrastive embedding for generalized zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 2371–2381.
[10] Y.-Y. Chou, H.-T. Lin, and T.-L. Liu, "Adaptive and generative zero-shot learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–14.
[11] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 951–958.
[12] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1778–1785.
[13] P. Morgado and N. Vasconcelos, "Semantically consistent regularization for zero-shot recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6060–6069.
[14] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2021–2030.
[15] S. Reed, Z. Akata, H. Lee, and B. Schiele, "Learning deep representations of fine-grained visual descriptions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 49–58.
[16] J. L. Ba, K. Swersky, and S. Fidler, "Predicting deep zero-shot convolutional neural networks using textual descriptions," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4247–4255.
[17] Y. Liu, L. Zhou, X. Bai, Y. Huang, L. Gu, J. Zhou, and T. Harada, "Goal-oriented gaze estimation for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 3794–3803.
[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, Jun. 2016, pp. 770–778.
[19] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1501–1510.
[20] D. Ruta, S. Motiian, B. Faieta, Z. Lin, H. Jin, A. Filipkowski, A. Gilbert, and J. Collomosse, "ALADIN: All layer adaptive instance normalization for fine-grained style similarity," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2021, pp. 11926–11935.
[21] H. Kazemi, S. M. Iranmanesh, and N. Nasrabadi, "Style and content disentanglement in generative adversarial networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 848–856.
[22] D. Kotovenko, A. Sanakoyeu, S. Lang, and B. Ommer, "Content and style disentanglement for artistic style transfer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4422–4431.
[23] Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4166–4174.
[24] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele, "Latent embeddings for zero-shot classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 69–77.
[25] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3174–3183.
[26] E. Kodirov, T. Xiang, Z. Fu, and S. Gong, "Unsupervised domain adaptation for zero-shot learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2452–2460.
[27] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong, "Transductive multi-view zero-shot learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 11, pp. 2332–2345, Mar. 2015.
[28] Z. Wan, D. Chen, Y. Li, X. Yan, J. Zhang, Y. Yu, and J. Liao, "Transductive zero-shot learning with visual structure constraint," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.
[29] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 819–826.
[30] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5327–5336.
[31] B. Romera-Paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2152–2161.
[32] S. Biswas and Y. Annadani, "Preserving semantic relations for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7603–7612.
[33] Y. Xian, S. Sharma, B. Schiele, and Z. Akata, "F-VAEGAN-d2: A feature generating framework for any-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10275–10284.
[34] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5542–5551.
[35] S. Narayan, A. Gupta, F. S. Khan, C. G. Snoek, and L. Shao, "Latent embedding feedback and discriminative features for zero-shot classification," in *Computer Vision*. Glasgow, U.K.: Springer, Aug. 2020, pp. 479–495.
[36] A. Mishra, S. K. Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2188–2196.
[37] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.
[38] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[39] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 172–189.
[40] B. Tong, C. Wang, M. Klinkigt, Y. Kobayashi, and Y. Nonaka, "Hierarchical disentanglement of discriminative latent features for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11467–11476.
[41] Z. Chen, Y. Luo, R. Qiu, S. Wang, Z. Huang, J. Li, and Z. Zhang, "Semantics disentangling for generalized zero-shot learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2021, pp. 8712–8720.
[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[43] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.
[44] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2623–2631.
[45] H. Jiang, R. Wang, S. Shan, and X. Chen, "Transferable contrastive network for generalized zero-shot learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9765–9774.
[46] W. Xu, Y. Xian, J. Wang, B. Schiele, and Z. Akata, "Attribute prototype network for zero-shot learning," 2020, *arXiv:2008.08290*.
[47] Y. Shen, J. Qin, L. Huang, L. Liu, F. Zhu, and L. Shao, "Invertible zero-shot recognition flows," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 614–631.
[48] Y. Yu, Z. Ji, J. Han, and Z. Zhang, "Episode-based prototype generating network for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 14035–14044.
[49] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-UCSD birds-200–2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.
[50] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning—A comprehensive evaluation of the good, the bad and the ugly," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2251–2265, Sep. 2019.
[51] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proc. 6th Indian Conf. Comput. Vis., Graph. Image Process.*, Dec. 2008, pp. 722–729.
[52] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
[53] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–27, 2008.

YOOJIN AN received the B.S. degree in police administration and convergence software from Dongguk University, South Korea, in 2020, where she is currently pursuing the M.S. degree with the Department of Artificial Intelligence. Her current research interests include computer vision and zero-shot learning.

SANGYEON KIM received the B.S. degree in computer science and engineering and the M.S. degree in artificial intelligence from Dongguk University, in 2019 and 2022, respectively. He is currently an AI Research Engineer with NAVER WEBTOON AI. His research interests include computer vision and zero-shot learning.

YUXUAN LIANG is currently pursuing the Ph.D. degree with the School of Computing, National University of Singapore. He was a Research Intern at Microsoft Research, JD.COM, and the Sea AI Laboratory. He has published over 20 papers in the refereed conferences and journals, such as KDD, TKDE, NeurIPS, WWW, IJCAI, AAAI, MM, and UbiComp. His research interests include deep learning, machine learning, and their applications in urban areas.

ROGER ZIMMERMANN (Senior Member, IEEE) received the B.S. degree in informatik from the University of Applied Sciences and Arts, Northwestern Switzerland, in 1986, and the M.S. and Ph.D. degrees in computer science from the University of Southern California, Los Angeles, CA, USA, in 1994 and 1998, respectively. He is currently a Professor with the School of Computing, National University of Singapore. His research interests include multimedia, networks, and spatio-temporal data management.

DONGHO KIM received the B.S. degree in computer engineering from Seoul National University, South Korea, in 1990, and the M.S. and Ph.D. degrees in computer science from the University of Southern California, Los Angeles, CA, USA, in 1992 and 2002, respectively. He is currently a Professor with the Dongguk Institute of Convergence Education, Dongguk University, South Korea. His research interests include distributed systems, networks, and security.

JIHIE KIM received the B.S. degree in computer science and statistics and the M.S. degree in computer science and statistics from Seoul National University in 1988 and 1990, respectively, and the Ph.D. degree in computer science from the University of Southern California, in 1996. She is currently a Professor with the Department of Artificial Intelligence, Dongguk University, Seoul, South Korea. Her research interests include machine learning, NLP, and knowledge-based reasoning.