



Class Representative Learning for Zero-shot Learning Using Purely Visual Data

Mayanka Chandrashekar¹ · Yugyung Lee¹

Received: 11 January 2021 / Accepted: 15 April 2021 / Published online: 1 June 2021
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2021

Abstract

The building of robust classifiers with high precision is an important goal. In reality, it is quite challenging to achieve such a goal with the data that are typically noise, sparse, or derived from heterogeneous sources. Thus, a considerable gap exists between a model built with training (seen) data and testing (unseen) data in applications. Recent works, including zero-shot learning (ZSL) and generalized zero-shot learning (G-ZSL), have attempted to overcome the apparent gap through transfer learning. However, most of these works are required to build a model using visual input with associated data like semantics, attributes, and textual information. Furthermore, models are made with all of the training data. Thus, these models apply to more generic contexts but do not apply to the specific settings that will eventually be required for real-world applications. In this paper, we propose a novel model named class representative learning (CRL), a class-based classifier designed with the following unique contributions in machine learning: (1) the unique design of a latent feature vector, i.e., class representative, represents the abstract embedding space projects with the features extracted from a deep neural network learned only from input images. (2) Parallel ZSL algorithms with class representative learning; (3) a novel projection-based inferencing method uses the vector space model to reconcile the dominant difference between the seen classes and unseen classes. This study demonstrates the benefit of using the class-based approach with CRs for ZSL and G-ZSL on eight benchmark datasets. Extensive experimental results suggest that our proposed CRL model significantly outperforms the state-of-the-art methods in ZSL/G-ZSL based image classification.

Keywords Zero-shot learning · Image classification · Transfer learning

Introduction

Deep learning technologies have received significant attention for large-scale image classification in the area of computer vision. The substantial requirements for such image classification tasks are the availability of a large amount of labeled data. Besides the limited amount of labeled data for supervised learning, we face severe challenges in applying image classification models to real-world problems, such as a lack of effectively transferring knowledge from one domain to another, or the effective adaption for newly generated data. There is a strong need for systematic research

exploring the processes that support effective transfer learning at finding a robust and feasible solution for real-world applications.

Recent efforts are focused on developing zero-shot learning (ZSL) [1] or few-shot learning (FSL) [2, 3] that aims to handle the challenges of image classification. The goal of ZSL is to recognize instances from unseen (target) categories by using external linguistic or semantic information through intermediate-level semantic representations from seen (source) categories [4–6]. These studies focus on effective transfer learning by fully leveraging information from pre-trained models. The central idea behind these studies is that known and unknown classes' auxiliary information can be used for image classification [7, 8]. It will allow them to learn from a few training examples (few-shot) or even without seeing them (zero-shot). However, it is required to map new instances (unseen) into a semantic space, which is aligned with the pre-trained models (seen). This will lead

✉ Mayanka Chandrashekar
mckw9@mail.umkc.edu

¹ University of Missouri-Kansas City, Kansas City, MO, USA

to more effective transfer of learning and help to minimize training efforts and maximize learning outcomes [9–11].

For the effective transfer learning from the known to the unknown, one of the most popular ZSL approaches is the semantic space model that is based on a joint embedding framework of the label-embedding space [1, 12] and the attribute space [13]. The label-embedding space is based on a combination of visual embeddings and word embeddings, while the attribute space is based on attribute annotations for the ZSL model. External or auxiliary information is used as a form of combining two or more sources of data, e.g., image features and word embedding, image features and attribute information, or image features and ontologies. We propose a novel ZSL model called class representative learning (CRL), which builds a class prototype from the instances of each class and defines it as a class representative (CR), which will be used for inference. The class prototypes for source and target domains are generated using the features extracted from a base architecture of the source (seen) domain. These class prototypes are used for inferencing with a projection function. CRs are a unified representation of the classes in the source and target domains, which support effective transfer-learning from the source to the target. Our study demonstrates the CRL's usefulness for transfer learning in image classification, with the significantly improved performance of ZSL and G-ZSL algorithms.

The class representative learning (CRL) model can be categorized as the projection method of Class-Inductive Instance-Inductive (as defined in [1]). In the training phase for the seen classes, the feature learning model is built from the training instances. During the testing phase for the unseen classes, both the unseen and seen prototypes are projected into the same space, based on learned models. Our study adopts the evaluation methods defined in Xian et al. [7, 8]. It extends it to the generalized zero-shot (G-ZSL) algorithm, in which the seen classes were built from the feature extraction learning from the ImageNet dataset [14]. The G-ZSL algorithm's performance was validated with the harmonic mean of seen and unseen classification performance [7, 8].

The contributions of this paper can be summarized as follows:

- Proposing the CRL model as an efficient way of building class-level classifiers by fully utilizing the features from a pre-trained Convolutional Neural Network (CNN) using visual data only;
- Designing a universal representation, called class representative feature space (CRFS), for source and target classes that can be applied to multiple cross domains;
- Applying the CRL model to ZSL and G-ZSL problems;
- Designing the parallel ZSL and G-ZSL algorithms based on CRL;

- Through extensive evaluations, the proposed CRL model shows significant performance gain compared to the state-of-the-art research in ZSL and G-ZSL.

The remainder of the paper is organized as follows. In "Related work", we literature review in ZSL and G-ZSL research. In "Class Representative Learning Model", we present the proposed CRL model and point out the justification of our design. In "Class Representative Inference", we describe the projection function in ZSL and present the ZSL/G-ZSL algorithms for the CRL inferencing. "Experiments" describes the design of the experiments. "Results and Evaluation" presents the results and makes comparisons with other models. Finally, in "Discussion" and "Conclusion", we provide a brief discussion and conclude the paper by presenting conclusions based on our work.

Related Work

Universal Representation

Some various research works are proposed to combine datasets to enhance models. Ubernet [15] is a universal CNN that allows solving multiple tasks efficiently in a unified architecture. It provides a simple end-to-end network architecture for diverse datasets, and scalable and efficient low memory processing. Also, universal representations [16, 17] have been shown to work well in a uniform manner for visual domains. They have proven to be efficient for multiple domain learning in relatively small neural networks. Rebuff et al. [18] presented that universal parametric families of networks could share parameters among multiple domains using parallel residual adapter modules. Similar to our work, all these works presented universal representations for various domains or various tasks. However, unlike CRL, their models cannot adequately support effective transfer learning in a distributed and parallel manner.

Feature selection is a crucial step in machine learning since it directly influences the performance of machine learning. The right choice of features drives the classifier to perform well. However, Kapoor et al. [19] observed that finding useful features for multi-class classification is not trivial. It is because of the volume in the high-dimensional feature space as well as the sparseness over the search space. Dictionary learning [20] was presented to determine the subspaces and build dictionaries by efficiently reducing dimensionality for efficient representations of classes in the domain. They overcame the sparsity constraints and improved the accuracy by identifying essential components of the observed data. In CRL, the class representative feature space CRFS provides a basis for building the prototypes, i.e., CRs, using the features from the CNN network that are

a uniform representation of the images. The proposed CRL model is based on distributed and parallel processing, which improves the efficiency of the CR generation and inferencing. It is possible due to the nature of the independence among CRs.

In the context of zero-shot learning and few-shot learning, the representatives are known as a class prototype, which is defined as vector representation in semantic space corresponding to each class [1]. EXEM proposed the class exemplar, which is similar to class representatives in CRL, as the center of visual feature vectors used in prediction and label embedding [21]. A hierarchical super prototype was proposed based on a combination of semantic prototype and visual data for seen and unseen classes [22]. Fu et al. proposed a prototype graph where each prototype is a node in the graph, and the semantic relationship between classes is an edge for their connectivity [23]. The prototype is also used in few-shot learning (FSL), as demonstrated by Snell et al. [24].

In the CRL model, the class prototype is part of the predictive step, unlike the previous works defined as an intermediate step towards a learning model. The class prototype, i.e., CRs, can be built independently class by class to be self-contained and is not dependent on other classes. For example, a class representative (CR) for dog class is built using the data only from one class (i.e., dog), regardless of other classes, such as horse or cat.

Zero-Shot Learning

A semantic encoding was derived from a semantic knowledge base [25] for predicting new classes in Zero-shot learning (ZSL). Besides the knowledge bases, explicit and external attributes are considered for visual learning [25]. In addition to standard image feature extraction techniques, other feature learning techniques such as boosting techniques [26], object detection [27], chopping algorithm [28], feature adaption [29], and linear classifiers [30] are used to enhance the accuracy for predicting unseen classes.

In Wang et al. [1], zero-shot learning was categorized, based on feature requirements, into *engineered semantic space* for attribute, lexical, and keyword information and *learned semantic space* for label-embedding,

text-embedding, and image-representation. Table 1 shows the CRL model and the state-of-the-art ZSL models which are compared in the evaluation section. Recent ZSL works mostly include two kinds of semantic spaces, namely *label-embedding spaces* [1, 12] and *attribute spaces* [1] (also known as probability prediction strategy [12]). The image representative space is present in the zero-shot learning works but it is typically coupled with an additional semantic space. The ZSL's label-embedding space focus on learning a projection strategy. This strategy maps semantic features extracted from the image representative space to the labels that are represented in a high dimensional embedding such as Word2Vec [31] or Glove [32]. Image representative space are typically learned from convolutional neural networks [13, 33–37]. Attribute space or Probability prediction strategy pre-trains attribute classifiers based on the source data [12], where an attribute is defined as a set of terms having the properties for a given class [1]. A class has distinguishing attributes, the embedding of visual features for the defining characteristics of the class, and assignment of the label to a class based on these features and attributes [4, 6, 38]. Unlike these works, CRL focuses on uses a mono-modal, specifically just the image representative semantic space. Using only the image representative gives CRL the advantage in terms of classification performance comparing to using a multi-modal semantic space approach.

ZSL approaches using only image representation space are rarely observed. One of the few works was based on the image deep representation (i.e., neural network-based) and fisher vector for the inference [39], and an extension of this approach was used to create unsupervised domain adaptation [40]. Zhu et al. used a partial image representation method to achieve a universal representation for action recognition [41]. Similarly, CRL uses only image representative semantic space. However, CRL utilizes a unique method of creating class prototypes and perform inference all within the space.

ZSL Projection Methods

The ZSL approaches can be categorized into classifier-based and instance-based methods. Based on the categorization, the CRL model is defined as a zero-shot learning model

Table 1 Related work: zero-shot learning methods

| | |
|-----------------------------|---|
| Instance-based ZSL method | |
| Projection method | CRL (ours), DWV [42], Deep-SVR [4, 43], ConSE [6], CMT [44], SAE [45], Embed [46] |
| Synthesizing method | GAN+ALE [47], GAN+Softmax [47], CADA-VAE [48], cycle-GAN [49], BPL+LR [50] |
| Classifier-based ZSL method | |
| Relationship method | SSE [51], AMP [35], SynC [52] |
| Correspondence method | SJE [33], LATEM [37], ALE [5], DeViSE [34], ESZSL [36], SP-AEN [53] |

that uses image representation semantic space and employs an instance-based projection method for model building and inference. The projection method provides insights on the labeled instances of an unseen class by projecting them onto the common feature space or the semantic space where instances and prototypes are compared [1]. A literature review is conducted as described below in the four different inference methods of zero-shot learning: *classifier-based correspondence method*, *classifier-based relationship method*, *instance-based projection method*, and *instance-based synthesizing method*.

Classifier-Based Correspondence Method constructs a correspondence between binary one-vs-rest classifier and unseen class prototypes to classify unseen classes. The compatibility function is a key part of the correspondence method. It takes instances and prototypes as an input to compute a compatibility score denoting the probability of instances to classes [1, 7, 54]. A bilinear function is a widely used compatibility function, including DeViSE [34], ALE [54], SJE [33], and ESZSL [36]. The other widely used method is the projection functions, such as linear projection [55, 56]. Even though the correspondence method uses classifier and prototype method, no explicit relationships between classes are modeled due to one vs. all strategy. As the class relationship plays a vital role in understanding ZSL performance, the CRL model uses a pure-prototype approach that aims to discover the class relationship to increase the model's interpretability.

Classifier-Based Relationship Method constructs classifier for the unseen classes based on the relationships among classes [1]. SSE [51] uses binary one-vs-rest classifiers for seen classes, and unseen class prototypes establish their relationship with seen classes. AMP [35] uses a directed k -nearest neighbor graph where each edge establishes the relationship between the classes. The relationship method focuses on creating a classifier by creating a class-to-class relationship; this inter-class dependency creates scalability issues. To handle scalability issues, CRL focuses on inter-class independence during model building, and the relationship between class is determined using their prototypes.

Instance-Based Projection Method classifies the instances of the unseen classes by projecting both the feature space instances and the semantic space prototypes into a shared space [1]. The projection space is defined as a space where the classification is performed. The approaches using projection methods are further categorized according to the projection space: *semantic space as projection space*, *visual space as projection space*, and *Transductive projection strategy*.

Semantic Space as Projection Space: A projection function is used to project the visual feature space into a linear or non-linear semantic space. The goal of cross-modal transfer (CMT) based on semantic word vector representations and Bayesian framework is to differentiate the semantic manifold of seen classes and unseen classes [44]. Some of these works use regression function as the projection method as well as softmax classifier on the semantic space [25, 44, 57]. A projection method was achieved from a convex combination of seen class prototypes (ConSE) [6], which has an n -way classifier built on seen data and is used to predict probabilities on the unseen instances. The projection function of Deep-SVR consists of attribute-based classifiers. The probabilities from attribute classifiers are coordinates with projected instances in semantic space using 1-NN classification [4, 43].

Visual Space as Projection Space: The projection function was learned to project the semantic space to visual feature space. Linear regression-based projection approach was used to mapping from the target space to the source space [58]. However, there is a strong assumption of a multivariate normal distribution of data and it lacks advanced similarity measures, such as cosine similarity or multi-modal data distributions. Non-linear regression (DEM) uses the visual space as the embedding space, resulting in fewer hubness problems than other ZSL approaches [46]. Unseen visual data synthesis (UVDS) was proposed with a latent embedding space that takes into account both semantic space and visual feature space [59]. Multiple projection spaces, such as semantic auto-encoder (SAE), were designed to learn a more generalized projection function [45].

Transductive Projection Strategy Manifold regularization was used together with data augmentation strategies to enhance the semantic space, which has resulted in easy access to testing data in the training phase [60]. Matrix factorization with testing instances and unseen class prototypes was designed for unsupervised domain adaptation to overcome the projection domain shift problem [61]. The self-training strategy aims to adjust the prototypes of unseen classes with the testing instances when performing 1-NN classification. For an unseen class, the prototype is adjusted as the mean value of the k nearest testing instances [62–66]. This unseen prototype creation is also used in a few-shot learning setting [24]. Markov chain process-based projection method was proposed to compute semantic manifold distance in embedding space as a seamless fusion of the semantic relatedness and embedding-based methods for ZSL [35]. Fu et al. presented a unified framework based on vocabulary-informed learning. It incorporates distance constraints from vocabulary atoms for projecting closer to

their correct prototypes in semantic manifold-based recognition [42].

As mentioned earlier, the CRL model can be categorized as a projection method. Unlike most of the existing works, CRL uses the visual feature space that is created by a projection function for inference. Class representative generation is similar to the prototype created by self-training strategy. The zero-shot learning with self-training uses the visual feature and additional semantic space for creating the prototype. The few-shot learning approach uses a massive network classifier model post prototype generation. The unique feature of CR generation is the sole use of visual feature space.

Instance-based synthesizing method of the zero-shot learning has been prevalent to a generative neural network in creating additional data points, especially for unseen classes, and handling the problem pertinent to data imbalance issues. GAN+ALE [47], GAN+Softmax [47] and cycle-GAN [49] rely on various types of generative adversarial network (GAN) model, which samples a random vector, combines that with the unseen class prototype to form the input to the generator, whereas CADA-VAE [48] uses variational auto-encoder for data synthesizing. BPL [50] uses semantic feature synthesis by perturbation approach that directly perturbs the seen class samples onto unseen class prototypes. In the context of the projection method, BPL [50] uses bidirectional projection learning as the part of a competitive learning strategy between seen samples and unseen prototypes. Synthesizing methods differ from the CRL model in architectural sense, but we compare it with CRL since it is known as one of the ZSL’s top-performing models.

Class Representative Learning Model

In this paper, we present the class representative learning (CRL) model designed to project the input data onto a global space and generate a universal representation for domains and use it for inference in zero-shot learning. The space of the CRL model is similar to the universal representation proposed by Tamaazousti et al. [67], where visual elements in the configuration (e.g., scale, context) can be encoded universally for the transfer learning. Unlike their work, our CRL representation is defined based on the aggregation patterns from the activation of neurons of the projection function (typically a pre-trained model, such as CNN). The CRL model’s fundamental concept is its ability to create the representatives of class independently from other classes for a given domain.

Figure 1a shows the conventional label-embedding based zero-shot learning model. Figure 1b shows the class-representative model in a similar setting. As shown in Fig. 1a, Nourouzi et al. introduced zero-shot learning with a two-step mapping function [6], where the first step is a projection function, and the second step is an inference function in the label-space. As shown in Fig. 1b, the two-step mapping stays in place, but the second mapping becomes non-essential because the class representatives (image feature prototypes) have the label as tagged information. In the label-based embedding model (Fig. 1a), the second mapping plays an essential role where the image feature space maps into the label space. In most of the existing works, the classification happens in the label space, wherein CRL’s classification happens only in an image feature space, also known as class representative feature space (CRFS) [68].

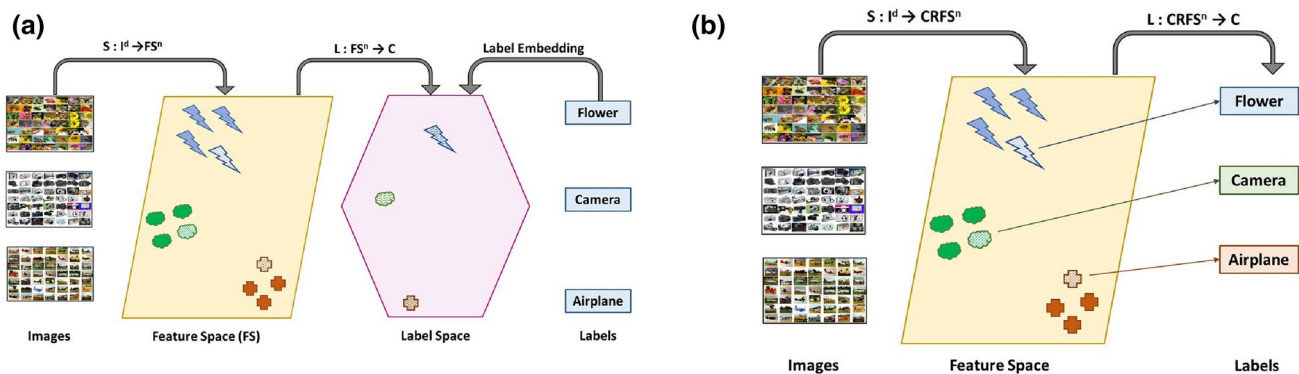


Fig. 1 a Label Embedding Based ZSL. b Class Representative Based ZSL

Table 2 Formal symbol and notations in the CRL model

| Notation | Description |
|-------------------------------|--|
| D_s and D_t | Source and target domain |
| m_s and m_t | #Data points from source and target, respectively |
| \mathbb{S} and \mathbb{T} | Source and target label set |
| C | #Classes |
| x | Feature vector of labeled data point |
| y | Label of data point |
| j | #Neurons in a given activation layer |
| b | Base vector learned in $P(\cdot) \{b_1, b_2, \dots, b_j\}$ |
| $CR(c)$ | Class representative of class c where $c \in C$ |
| x^* | Unlabeled data point |
| y^* | Predicated label for x^* |
| $CRC(\cdot)$ | Class representative classifier |
| $P(\cdot)$ | Projection function |
| $L(\cdot)$ | Inference function |
| $CRFS^n$ | Class representative feature space |
| n | Dimensions of the $CRFS^n$ |

Problem Setup

Assume the given source data $D_s = \{x_i, y_i\}_{i=1}^{m_s}$ of m_s labeled points with a label from the source class \mathbb{S} , where $x_i \in \mathbb{R}$ is the feature of the i^{th} image in the source data and $y_i \in \mathbb{S}$, \mathbb{S} is the set of source classes. The target data is represented as $D_t = \{x_i, y_i\}_{i=1}^{m_t}$ classes where $y_i \in \mathbb{T}$. Each class $c \in \mathbb{S} \cup \mathbb{T}$ has a class representative $CR(c)$, which is the semantic representative of class c . Furthermore, the source label \mathbb{S} and the target label \mathbb{T} are considered such that $\mathbb{S} \cap \mathbb{T} = \emptyset$. For simplicity, the source and target datasets may have overlapped labels, but these overlapped classes are considered distinct. In the CRL model, the source data are considered as *seen*, and the target data are *unseen*. In other words, the target data are not used in the learning process. Table 2 summarizes all the symbols and notations used in the CRL model. The goal of the CRL model is that given a new test data x^* , the model classifies it into one of the classes y^* , where $y^* \in C$. The CRL model defines a universal problem for a ZSL approach as well as G-ZSL as follows:

- Zero-shot learning (ZSL): $y^* \in \mathbb{T}$
- Generalized zero-shot learning (G-ZSL): $y^* \in \{\mathbb{S} \cup \mathbb{T}\}$

There are no dependencies among CRs in any of the models. The difference between these two models is in the properties of the inference. If the CR of the target set was introduced, then it would be ZSL, and if both CRs of the source set and the target set are introduced, then it would be G-ZSL.

Definition: Class representative (CR) is a representative of K instances in a single class. The activation feature map of the CR is a unique characteristic pattern of visual expression using the projection function; for example, the feature map is generated from the deep learning process using convolutional neural networks as the projection function (CNN). Activation feature map (AFM) is a vector of features extracted from a base model, which is learned from the source dataset. Thus, CR is an abstraction of instances of a class by computing an aggregation of the average mean vectors of the AFM for the K instances. The CR represents the characteristics of a class as well as it differentiates one class against another. The class representatives $CR(c)$ for Class c is represented as $\{CR_c^1, CR_c^2, \dots, CR_c^n\}$ with n dimensional aggregated features. Each dimension corresponds to a separate feature. If a feature occurs in CR, its value in the vector is non-zero.

$$CRC = L(P(\cdot))$$

$$P : I^d \rightarrow CRFS^n \quad (1)$$

$$L : CRFS^n \rightarrow c$$

Class Representative Classifier

We define a class representative classifier $CRC : I^d \rightarrow c$ that maps an input image space I^d of the dimension d and $c \in \mathbb{S} \cup \mathbb{T}$. Classifier has two essential functions, such as the projection function $P(\cdot)$ and inference function $L(\cdot)$. The projection function takes the input images and learns the feature space for class representatives through source data D_s . The inference function uses the class representative feature space $CRFS^n$ of the dimension n to classify it to class c . CRC is defined as a composition of these two functions, as shown in Eq. 1.

The CRC model first learns a projection function S using the seen data, which aids the mapping of the input images I^d into class representative feature space $CRFS^n$ with n dimensions. \mathbb{T} the inference function L creates class representatives CR in the $CRFS^n$ of either the seen data or unseen data depending on the setting. The L maps a new image to Label c in $CRFS^n$ where $c \in \mathbb{T}$ or $c \in \mathbb{S} \cup \mathbb{T}$.

Projection Function

In the CRL method, we use a projection mechanism that uses only a visual feature space. The visual feature space in the CRL method is known as the class representative feature space. The projection function is prevalent as part of the class representative classifier in two ways. First, learning the projection method using the seen data and building the class representative feature space $CRFS^n$. Second, applying the pre-learned projection mapping mechanism to the seen

data or the unseen data depending on the setting for inference (e.g., ZSL or G-ZSL).

CRFS is defined as a n dimensional semantic feature map in which each of the n dimensions represents the value of a semantic property. These properties may be categorical and contain real-valued data or models from deep learning methods [25]. Class representative feature space is the projection space. The class representative feature space represents n dimensional representative features as a form of the activation feature map (AFM). The design of the CRFS is based on the equations defined in [69]. The data points from D_t $\{(x_1, y_1, \dots, (x_m, y_m))\}$ with each feature of $x_i \in \mathbb{R}$ and $y_i \in \mathbb{T}$ (as shown in Eq. 3). Note that the data points can also be defined in D_s that is used in CRFS of both source and target domains (refer to "Validation using Domain Adaptation").

$$\hat{a}(x_i) = P(x_i)$$

$$\hat{a}(x_i) = \underset{(a^{(i)})}{\operatorname{argmin}} \|x_i - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1 \quad (2)$$

where $D_s = \{x_i, y_i\}_{i=1}^{m_s}$

$$D_s \xrightarrow{P_b} \hat{D}_s \quad (3)$$

$$\hat{D}_s = \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_s}$$

Class representative feature space is created based on the base vector b , which has j dimensions with each $b_j \in R_n$. The base vector b is generated with the projection function using D_s as an optimization function like stochastic gradient descent. Equation 2 points out the self-training based on unsupervised data. The activation $\hat{a}(x_i)$ consists of $\{\hat{a}(x_i^1), \dots, \hat{a}(x_i^n)\}$ with each $\hat{a}(x_i) \in \mathbb{R}$ forms the CRFS's semantic property. Each dimension of an activation vector $\hat{a}(x_i)$ is the transformation of input x_i using the base b_j , where j is independent of size of input x_i . The base b is learned through a smooth approximation of L_1 sparsity penalty $a(i)$. Even though the approximation does not lead to sparse features, Raina et al. used re-calibrated β value before computing a labeled data representation and improved the classification accuracy [69]. This particular base vector definition we used as the projection function $P(\cdot)$ is not dependent on the supervised nature of neural network models typically used. For evaluation purposes, we consider the base b as learned based on one of the convolution network layer, where b is generated based on the learning on the seen data. Projection

function $P(\cdot)$ can potentially be mapped to a convolution network layer or a residual network layer. The advantage of having a projection function purely for mapping; any pre-trained models can be used in it is place. The pre-trained model does not influence the ability of the projection function mapping, instead of contributing to the optimization of the base vector. The projection function, independently learned from the source classes, is playing an integral role in the CRL model.

Class Representative Generation

Class representatives (CR) are generated using the nearest prototype strategy by aggregating feature vectors. As the name specifies, class representatives create representatives from the instances projected in class representative feature space $CRFS^n$ using Projection Function $P(\cdot)$. Class representatives (CR) and the instances share the same feature space $CRFS^n$. The nearest mean feature vector with the instances of the given class, i.e., class representatives, is computed for every class. Correctly, the average feature mean operation was used to summarize the instances of classes. The CR is an aggregated vector of the mean features for all the elements in the feature maps.

For the CR generation, we considered the transformed source dataset \hat{D}_s as the input (as shown in Eq. 2). As we emphasize on the distributed and parallel processing with CRs, we considered the individual activation vector $\hat{a}(x_i)$ such that $y_i = c$ where $c \in \mathbb{S}$, that will be used in formulating the CR generation as shown in Eq. 4.

$$\hat{D}_s = \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_s}$$

$$CR(c) = \begin{cases} CR^1 = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^1), & \text{if } y_i = c \\ CR^2 = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^2), & \text{if } y_i = c \\ \vdots \\ CR^n = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^n), & \text{if } y_i = c \end{cases} \quad (4)$$

$$CR(c) = \{CR^1, CR^2, \dots, CR^n\}, \quad \forall c \in \mathbb{S}$$

Equation 4 shows the feature-wise CR generation in CRFS. For the CR generation, the projected source data \hat{D}_s is considered. For each class $c \in \mathbb{S}$, the projected source data $\{(\hat{a}(x_i), y_i)\}$ is considered such as $y_i = c$. Each feature is represented in the $CRFS^n$ dimension ranged 1 to n , the average represents the corresponding dimension for $CR(c)$.

Class Representative Inference

Zero-Shot Learning Setting

Algorithm 1: CRC-Inference: Zero-Shot Learning Setting

```

Input:  $D_t = \{x_i, y_i\}_{i=1}^{m_t}; x^*$ 
Output:  $y^*$ 
ZSL Setting:  $y^* \in \mathbb{T}$ 
Projection Function  $P_b(x)$ :
    /* Base Vector  $b$  was learnt from  $D_s$ , Based on Equation 2 */
     $\hat{a}(x) = P(x)$ 
    return  $\hat{a}(x)$ 
Inference Function  $L(CR(c) \forall c \in \mathbb{T}, \hat{a}(x^*))$ :
     $y^* = \operatorname{argmax}_{c \in \mathbb{T}} \{\cos(CR(c), \hat{a}(x^*))\}$ 
    where
    
$$\cos(CR(c), \hat{a}(x^*)) = \frac{CR(c) \cdot \hat{a}(x^*)}{\|CR(c)\| \|\hat{a}(x^*)\|}$$

    
$$= \frac{\sum_{j=1}^n \{CR^j(c) * \hat{a}(x^{j*})\}}{\sqrt{\sum_{j=1}^n (CR^j(c))^2} \sqrt{\sum_{j=1}^n \hat{a}(x^{j*})^2}}$$

    return  $y^*$ 
CR-Classifier Function  $CRC(D_t, x^*)$ :
    /* Projection of Unseen Data  $D_t$  into  $CRFS^n$  Similar to Equation 3 */
     $D_t \xrightarrow{P_b} \hat{D}_t$  for  $i \in 1 \rightarrow m_t$  do
     $\hat{a}(x_i) = P_b(x_i)$ 
     $\hat{D}_t = \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_t}$ 
    /* Projection of New Data Point  $x^*$  into  $CRFS^n$  */
     $\hat{a}(x^*) = P_b(x^*)$  /* CR Generation: Based on Equation 4 */
    for  $c \in \mathbb{T}$  do
    
$$CR(c)^j = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{a}(x_i^j),$$

    if  $y_i = c$  &  $\forall j \in (1, n)$ 
    /* Inference Function */
     $y^* = L(CR(c) \forall c \in \mathbb{T}, \hat{a}(x^*))$ 

```

The ZSL setting for class representative classifier (CRC) is designed with the pre-learned projection function P of the source or seen dataset D_s on the unseen or target dataset D_t . The CR-based classifier involves the following three steps: projection function (as describes in "Projection Function"), class representative generation (as described in "Class

Representative Generation"), and inference function L . Algorithm 1 presents the projection function P_b that is pre-trained on the source dataset D_s and used to map the target D_t instances and the test data point x^* to class representative feature space $CRFS^n$. As is true for the projection methods, each unseen class needs at least one labeled instance

to create a prototype, i.e., class representative [1]. The class representative creation is independent of the number of instances per class, with the only requirement of having at least one labeled instance per class.

The aggregation of projected unseen dataset \hat{D}_t creates the class representative for all classes $c \in \mathbb{T}$. As shown in Eq. 4, the dimensions of the projected are maintained for the class representative as well. The class representatives $CR(c) \forall c \in \mathbb{T}$ and the new data point $\hat{a}(x^*)$ reside the same feature space, i.e., class representative feature space $CRFS^n$. For the inference, the cosine similarity between the class representative in T and projected new data point $\hat{a}(x^*)$ is calculated. The label information is retained and coupled with each class representative. For the final inference step, the label of the class representative with the highest cosine similarity is returned as the predicted class y^* for the new data point x^* . The CRC classifier uses an instance-based projection method and inference method with no learning for unseen data or any new data as an ideal zero-shot learning setting.

Generalized Zero-Shot Learning

Algorithm 2 shows the variation of Algorithm 1, incorporating the generalized zero-shot learning(G-ZSL) setting. In the G-ZSL setting, the source dataset, target dataset, and new data point are projected into the feature space for the projection function. The projection method used here is based on pre-trained base vectors b learned from the source or seen dataset. Having projected both seen and unseen dataset onto the class representative feature space $CRFS^n$, class representatives are generated for all class $c \in \mathbb{S} \cup \mathbb{T}$ as shown in Eq. 4. The class representative $CR(c) \forall c \in \mathbb{S} \cup \mathbb{T}$ and the projected data point $\hat{a}(x^*)$ reside in the same feature space $CRFS^n$ for the inference step. The inference step involves getting c , where the class representative $CR(c)$ has the highest cosine similarity with the data point, leading to the conclusion that data point is predicted with the label c .

Algorithm 2: CR-Inference: Generalized ZSL Setting

Input: $D_s = \{x_i, y_i\}_{i=1}^{m_s}$; $D_t = \{x_i, y_i\}_{i=1}^{m_t}$; x^*
Output: y^*
G-ZSL Setting: $y^* \in \{\mathbb{S} \cup \mathbb{T}\}$
CR-Classifier Function $CRC(D_s, D_t, x^*)$:

```

/* Projection of Unseen Data  $D_t$  into  $CRFS^n$  Similar to
Equation 3 */
 $D_t \xrightarrow{P_b} \hat{D}_t$ 
/* Projection of Seen Data  $D_s$  into  $CRFS^n$  Similar to
Equation 3 */
 $D_s \xrightarrow{P_b} \hat{D}_s$ 
/* Projection of New Data Point  $x^*$  into  $CRFS^n$  */
 $\hat{a}(x^*) = P_b(x^*)$  // CR Generation: Based on Equation 4
for  $c \in \{\mathbb{S} \cup \mathbb{T}\}$  do
     $CR(c)^j = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{a}(x_i^j),$ 
    if  $y_i = c$  &  $\forall j \in (1, n)$ 
// Inference Function
 $y^* = L(CR(c) \forall c \in \{\mathbb{S} \cup \mathbb{T}\}, \hat{a}(x^*))$ 

```

Model Parallelism

Algorithm 3: CRC-Inference: ZSL Setting [Parallel Mode]

Input: $D_t = \{x_i, y_i\}_{i=1}^{m_t}; x^*$

Output: y^*

ZSL Setting: $y^* \in \mathbb{T}$

Inference Function $L(CR(c) \forall c \in C_p, \hat{a}(x^*))$:

$$y_p = \operatorname{argmax}_{c \in C_p} \{\cos(CR(c), \hat{a}(x^*))\} \quad (8)$$

/ score_p is the cosine similarity between CR(y_p) and x* */*

return (y_p, score_p)

CR-Classifier Function $CRC(D_t, x^*)$:

Broadcast:

Pre-trained base vectors b and new data point x^* */* C_p represents subset of classes split based on distribute function. D_p is data point for C_p with size of m_p */*

$D_p, C_p = \text{distribute}(D_t, \mathbb{T})$

$Y, SCORE = \{\text{empty}\}$

PARALLEL MODE

in parallel do

/ Projection of Distributed Data D_cs into CRFSⁿ Similar to Equation 3 */*

$D_p \xrightarrow{P_b} \hat{D}_p$ */* Projection of New Data Point x* into CRFSⁿ */*

$\hat{a}(x^*) = P_b(x^*)$ */* CR Generation: Based on Equation 4 */*

for $c \in C_p$ **do**

$$CR(c)^j = \frac{1}{m_p} \sum_{i=1}^{m_p} \hat{a}(x_i^j),$$

if $y_i = c$ & $\forall j \in (1, n)$

/ Classifier Function */*

$(y_p, score_p) = L(CR(c) \forall c \in C_p, \hat{a}(x^*))$

$Y, SCORE \leftarrow y_p, score_p$

end

REDUCTION STEP

/ Reduction Step combines results generated from each C_p */*

$$y^* = \operatorname{argmax}_Y \{SCORE\} \quad (9)$$

Algorithm 3 describes the parallelized zero-shot learning inference for the class representative classifier. As there is no learning for the target classes and no dependence between classes, CRC can support parallel processing for even larger datasets. The algorithm is designed with the CRCW (concurrent read concurrent write) model, which allows the parallel computing, including I/O, with the shared memory and

processors. The parallelized class representative classifier function CRC is composed of the following major steps:

First, involves in the declaration of the global variables, broadcasting certain inputs, and distribution of other inputs. Global variables are the variables that can be accessed and updated across the parallel process. Algorithm 3 declares predicted label set Y and corresponding cosine similarity score set $SCORE$ as global variables. The broadcast

variables are the ones that represent the new projected data point $\hat{a}(x^*)$ and pre-trained base vectors b for the projection function $P(\cdot)$.

Second, the target data points D_t and the target domain \mathbb{T} gets distributed into smaller sets of classes using *distribute(.)* function. The *distribute* function splits the target classes \mathbb{T} and the corresponding data points D_t into smaller sets of classes C_p and their corresponding data points D_p . The distribution can be aligned according to the CRL’s parallelization capacity. The class set C_p can be as small as a single class due to the independence between classes in target classes $c \in \mathbb{T}$. Each processor works with each set of data points D_p and classes C_p to create class representatives $CR(c)$ for all classes in C_p and compare with the new data point $\hat{a}(x^*)$. At the end of each parallel execution, a predicted label y_p and the cosine similarity $score_p$ between $CR(y_p)$ and $\hat{a}(x^*)$ are returned. The predicted label and score are accumulated onto the global variables, Y , and $SCORE$. The final resultant label y is obtained using the reduction step where the label with the highest cosine similarity in Y , $SCORE$, is determined (refer to Eq. 9 in Algorithm 3).

Validation using Domain Adaptation

We validate the domain compatibility by checking the compatibility between source (seen) domain and target (unseen) domain inspired from the existing domain adaptation techniques [70, 71]. The domain adaptation problem arises when the source domain data distribution is different from target domain data distribution. Domain adaptation aims to learn a predictor function in given a feature space using the source domain and apply it to the target domain. The hypothesis of domain adaptation is verified by measuring the distance between using the probability distribution obtained as the conditional probabilities of the outcome given predictor between source and target [70, 71].

In this paper, we use domain adaptation to validate the class representative distribution of source and target domains. The domain adaptation typically is used with the predictor function and the instance distribution of source and target to measure the divergence. Unlike this conventional way, we formalize the domain distribution using the cosine similarity between CRs of the sources classes and the target classes. The domain distribution using cosine similarity between CRs determine the degree of the similarity between the classes, i.e., if the current feature space is favorable for the discrimination of classes within the source and target domains. In this way, we can determined if the target domain has the same CR-to-CR cosine similarity distribution as the source domain.

Figure 2 shows an abstract three dimensional (X,Y,Z) vector space model of the CR cosine similarity distribution. Note that the dimensions of vector space model corresponds

to the n dimensions of CRFS. We consider three classes (i,j,k) from source domain and their corresponding class representatives (CR_i, CR_j, CR_k) . We also consider three classes $n(i',j',k')$ from target domain and their corresponding class representatives $(CR_{i'}, CR_{j'}, CR_{k'})$. The cosine similarity between each pair of class representative is calculated to each domain distribution in the class representative feature space (refer to Eq. 5).

$$f(\mathbb{S}) = \{\theta_{ij}, \theta_{ik}, \theta_{kj}\}$$

$$f(\mathbb{T}) = \{\theta_{i'j'}, \theta_{i'k'}, \theta_{k'j'}\} \tag{5}$$

where $\theta_{i,j} = \cos(CR_i, CR_j)$

The distribution differences over the instances in a given domain (i.e., either source or target) [70, 71] can be used for improving the target learner performance as the domain adaptation process. The distribution is the probability distribution over the prediction function; instead, we use the distribution of cosine similarity. ESZSL in [36] firstly defined zero-shot learning as a domain adaptation problem with the probability distributions over the instances. Romera et al. presented the theoretical model using \mathcal{A} -distance as the measurement between the source and target distributions. The \mathcal{A} -distance is defined as the total variation or the L^1 norm between the distribution within a given measurable subset \mathcal{A} with the domains [72].

Intuitively, \mathcal{A} -distance shows the most substantial change in the similarity of a given set. The set can be considered as a manual choice or conditional filtering, and we show conditional filtering set in Table 4. The first conditional set is obtained by filtering the higher cosine similarity $\forall(0.5 \leq \theta)$ that indicates the highly similar CRs, leading to the potential mis-classification between the classes. Similarly, the second

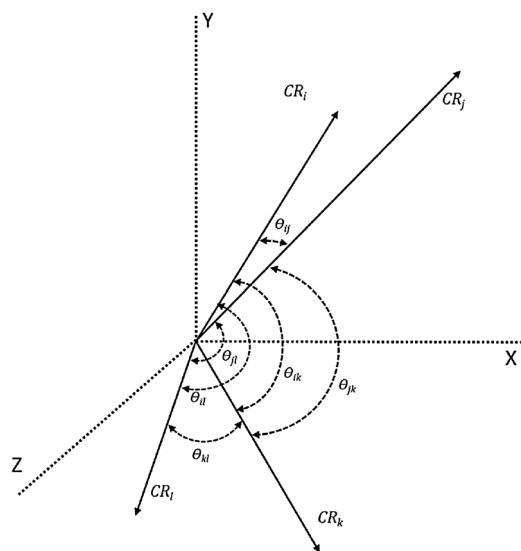


Fig. 2 Source domain—Cosine similarity distribution

conditional set is generated by filtering lower cosine similarity ($0 < \theta < 0.5$), which indicates the class representative pair with the least similarity.

$$\begin{aligned} V_{ks} &= \sup |f(\mathbb{S}) - f(\mathbb{T})| \\ V_{\mathcal{A}} &= 2 \sup_{\mathcal{A}} |f(\mathbb{S}_{\mathcal{A}}) - f(\mathbb{T}_{\mathcal{A}})| \end{aligned} \quad (6)$$

The Kolmogorov-Smirnov (*KS*) test is used to measure variation across the entire distribution, whereas *A*-distance is used to measure variation across a subset obtained from both the distribution for a given condition. As shown in Eq. 6, $f(\mathbb{S})$ and $f(\mathbb{T})$ are the distribution functions based on the cosine similarity distribution for the source and target domain, respectively. V_{ks} defines the Kolmogorov-Smirnov distance between the entire source CR distribution and target CR distribution. $V_{\mathcal{A}}$ defines the *A*-distance between the source and target domain on a given subset of data. For $V_{\mathcal{A}}$ and V_{ks} , a lower score means the target is closer to the source, and the higher score means the targets are further away from the source. The higher score indirectly indicates incompatibility between the target and the source, and the classification with the target might perform poorly because of the far distance, i.e., with a lower accuracy.

Experiments

Datasets

The CRL model was evaluated using six different target (unseen) datasets in two different settings. For our experiments, ImageNet-1K (2015), with 1000 classes [14], was used as the source dataset. The target dataset info is shown in Table 3. We considered four standard zero-shot learning (ZSL) datasets and two classification datasets for the target dataset. The four ZSL datasets include Animals with Attributes-2 (AWA2), Caltech-UCSD Birds-200 (CUB200), Scene Attribute dataset (SUN_A), and ImageNet-360 (IN360). IN360 includes 360 unique classes present in ImageNet-1K (2010) version, which is different from the source domain dataset, ImageNet-1K (2015) [42, 50, 63]. In addition, there

are the two classification datasets, including Caltech-101 (C-101), Caltech-256 (C-256).

Setting-1: We investigated the effects of the seen and unseen split for Setting-1 similar to work presented by Xian et al. [8]. The seen and unseen data in Setting-1 are prepared for a fair comparison with the state-of-the-art ZSL models. It is noted that the seen and unseen split is mainly for evaluation purposes. We want to highlight that there is no training activity happening with the unseen data in the CRL model since the ImageNet-1K pre-trained models are used as projection function as a default method (refer to "[Pre-trained Model as Projection Function](#)"). The ImageNet-1K pre-trained model should be sufficient for inference with the seen and unseen data for this setting.

Setting-2: The Setting-2 is a unique setup for CRL, where the source domain (ImageNet-1K) included in the pre-trained model is considered as seen classes. Typically, the G-ZSL setting includes both source and target classes as a class set during inference. For this G-ZSL setting, we argue that ImageNet-1K should be included with the given dataset in the class set as it is present during the learning of projection function (as we are using a pre-trained model). As CRL focuses on no learning for unseen classes, Table 3 Setting-2 shows the ImageNet-1K as the seen classes (source domain) and each dataset's categories as the unseen classes (target domain).

Pre-Trained Model as Projection Function

As described in "[Projection Function](#)", the projection function $P(\cdot)$ mapping the input images into the class representative feature space. For the experiments, we use widely available image classification models that are pre-trained using ImageNet-1k (2015). MATLAB's pre-trained deep neural networks [77] were used as projection functions, including Inception-V3 [78], ResNet101 [79], VGG-19 [80] and GoogleLeNet [81]. One of the CNN models was defined as the projection function (pre-trained with ImageNet-1K) for our experiments. The last convolution layer from the CNN model was considered as the base vector b . The j dimensions are based on the input size of the layer, and the layer's output

Table 3 Benchmark dataset: seen and unseen classes

| Dataset | #Class | #Image | Setting-1 | | Setting-2 | |
|-------------|--------|----------|-----------|--------|-----------|--------|
| | | | Seen | Unseen | Seen | Unseen |
| C-101 [73] | 101 | 8677 | – | – | 1000 | 101 |
| C-256 [74] | 256 | 30,608 | – | – | 1000 | 256 |
| AWA2 [8] | 50 | 30,475 | 40 | 10 | 1000 | 50 |
| CUB200 [75] | 200 | 11,788 | 150 | 50 | 1000 | 200 |
| IN360 [14] | 360 | 2,44,800 | 1000 | 360 | 1000 | 360 |
| SUN_A [76] | 806 | 14,340 | 697 | 109 | 1000 | 806 |

dimensions become n dimensions of class representative feature space CRFS.

$$\text{Acc}_T = \frac{1}{\|T\|} \sum_{c=1}^{\|T\|} \frac{\# \text{ correct predictions in } c}{\# \text{ samples in } c} \quad (7)$$

Evaluation Metrics

For zero-shot learning setting ("Problem Setup"), we use the average of class-wise accuracy Acc_T and flat-hit@ k . Equation 7 shows that class-wise accuracy is calculated by the average of correct predictions for each class. This evaluation was used to interpret the accuracy of the best performing class and the worst-performing class. Flat-hit@ k evaluation is defined as the percentage of the test images for which the model returns the matched label in its top k predictions. It is useful to determine whether the CRFS is crucial to get better accuracy by considering k nearest class representatives to a given instance.

$$H = \frac{2 * \text{Acc}'_S * \text{Acc}'_T}{\text{Acc}'_S + \text{Acc}'_T} \quad (8)$$

$$\text{Acc}'_S : \mathbb{S} \Rightarrow \mathbb{S} \cup \mathbb{T}$$

$$\text{Acc}'_T : \mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$$

For the G-ZSL setting, we used the harmonic mean (H) of the source dataset accuracy (Acc'_S) and the target dataset accuracy (Acc'_T) defined in [7, 8]. Acc'_S is calculated by considering the label space to be both source and target ($\mathbb{S} \cup \mathbb{T}$). Similarly, Acc'_T is calculated by considering correct predictions of target instances (Eq. 8).

Model Parallelism and System Specifications

A parallel processing is performed to extract features class by class and build a CR for each category ("Model Parallelism"). The CR generation was implemented parallel with Spark's resilient distributed datasets (RDDs), a collection of data points partitioned across the nodes of the cluster. The distribute function was implemented as the RDD partition with the condition that all data points of a given class are present in the same partition. The CR generation was implemented in the map stage, where each partition is independent of each other.

The pre-trained projection method was implemented on a single GPU, which is Nvidia GeForce GTX 1080 (with 12 GB GDDR5X RAM) on MATLAB 2018b version. The CR generation and CR-based inference were implemented using Spark 2.4.3 version [82]. The parallel and batch process was conducted through the RDD based parallelism on a single CPU with 4 GHz Intel Core i7-6700K (quad-core,

8MB cache, up to 4.2 GHz with Turbo Boost) and 32 GB DDR4 RAM (2133 MHz) (i.e., local parallelism of 4 cores).

Results and Evaluation

Domain Adaptation

The Kolmogorov-Smirnov Test (KS-Test) and \mathcal{A} -distance are used to identify the type of the transfer learning [83] happened during the zero-shot learning process from a given source dataset and target dataset. For this experiment, Inception-V3 based CRs were considered. Each CR was generated using ten labeled instances. Homogeneous transfer learning happens when the source and target feature spaces have the same attributes, labels, and dimensions. In this experiment, ImageNet-1K is identified as homogeneous transfer learning with V_{ks} and $V_{\mathcal{A}}$ scores as zero. If the source is identical to the target, it is not typically considered as transfer learning. The CRL model takes just the feature space established based on a pre-trained projection method but it uses a inference method that is not softmax. Thus, the CRL model in ImageNet-1K is identified as homogeneous transfer learning.

Heterogeneous transfer learning happens when the source and target domains share limited or no features or labels, and dimensions of the feature space differ as well. All the target domains can be considered as the heterogeneous transfer learning when no or little label overlap with the source domain, i.e., ImageNet-1K. The critical observation is if the heterogeneous transfer learning negatively impacts the target domain performance, that brings the issue of negative transfer. Negative transfer learning happens when the target domain's performance has negative implications on knowledge transfer from the source domain. The negative transfer learning is generally found when the source domain has the minimal similarity with the target domain. The KS-test and \mathcal{A} -distance is used to identify if the target domain has a negative transfer.

With the highest score in all variations i.e., V_{ks} and both $V_{\mathcal{A}}$, the dataset CIFAR-100 has the negative transfer. Figure 3 shows the target domain data are projected on the semantic space that is quite distinct from the source domain. Although the CIFAR-100 is semantically relevant to other datasets, the CRL space of CIFAR-100 is divergent from the source space in terms of image modality, such as image quality and image size. The size of CIFAR-100 images is [32 × 32] while one of the source domain ImageNet-1K is [400 × 400]. More specifically, the dimension of the projection method Inception-V3 is [299 × 299].

Figure 4 demonstrates the similarity distribution in the feature space of the source and target datasets. This figure further confirms the existence of negative transfer. The next

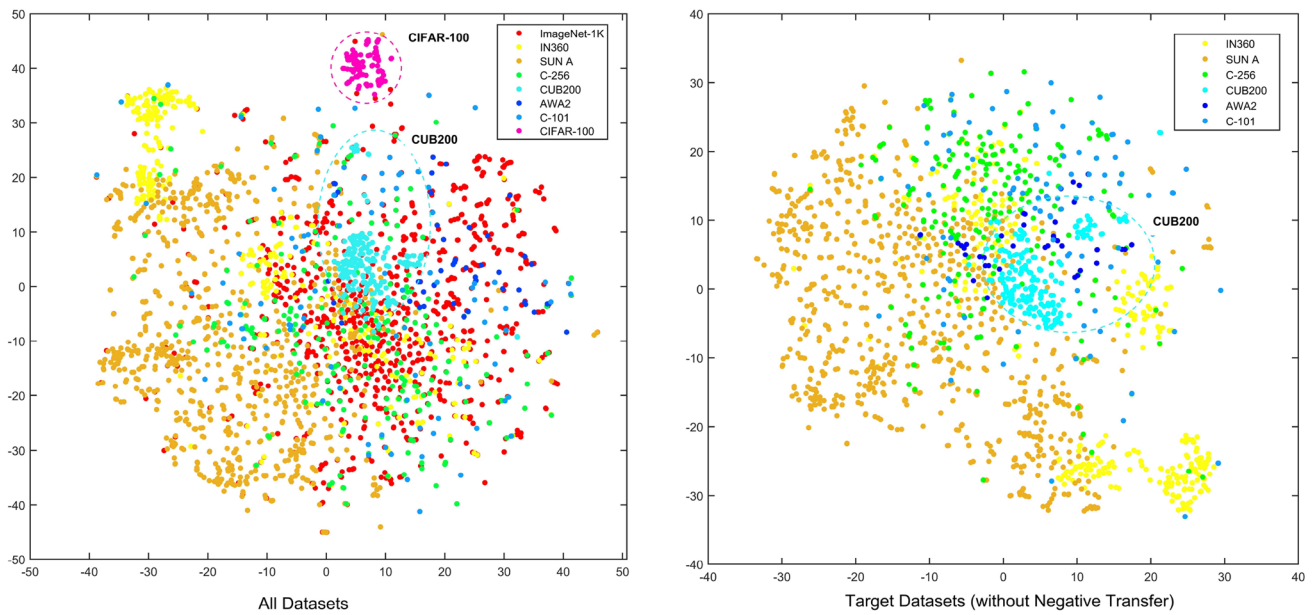
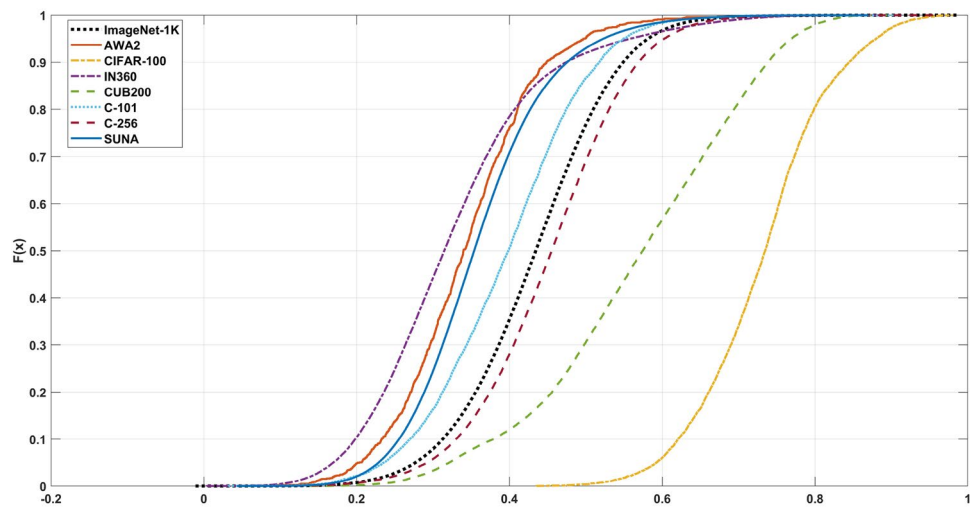


Fig. 3 t-SNE visualization of class representatives

Fig. 4 Cosine similarity distribution of the benchmark datasets



dataset which might have negative performance is CUB200, which a score of 0.9737 on cosine similarity greater than 0.5. This indicates that the distribution created by the high similar class representatives in CUB200 is very different from one in ImageNet-1K. This distribution disparity can also be observed in Fig. 3, where CUB200 forms a highly dense cluster in all the figures. It supports the nature of the dataset as CUB200 is specific to bird categories, whereas ImageNet-1K has a variety of animate and inanimate objects.

Table 4 shows that the accuracy of datasets does not perfectly correlate with the scores V_{k_s} and V_A . This lack of correlation is due to the size of the dataset i.e., several classes.

Figure 5 shows the class-wise distribution of the dataset, and the star marker indicates the number of classes. The SUN_A dataset's performance is dependent on multiple factors such as the size of the dataset, the standard deviation of the class-wise accuracy, and the domain compatibility score. The standard deviation is class-wise accuracy, and the significant difference between two V_A can be correlated. This inconsistency is due to a lack of quality images for certain classes. Figure 6 shows an example of two best performing and two worst-performing classes.

Table 4 Source and target domains: accuracy, kolmogorov-smirnov test scores, \mathcal{A} -distance high KS test score indicates negative transfer learning

| Target domain Dataset | Source and target comparison Acc_T (%) | Source and target comparison | | |
|--------------------------|---|------------------------------|---|--|
| | | V_{ks} | $V_{\mathcal{A}} \forall \theta(0.5 \leq \theta)$ | $V_{\mathcal{A}} \forall \theta(0 < \theta < 0.5)$ |
| ImageNet-1K | 73.7 | 0 | 0 | 0 |
| C-256 | 70.5 | 0.0921 | 0.0908 | 0.1132 |
| C-101 | 91.2 | 0.1570 | 0.1350 | 0.2767 |
| SUN_A | 31.9 | 0.3560 | 0.1398 | 0.6205 |
| AWA2 | 76.8 | 0.4120 | 0.2848 | 0.7228 |
| IN360 | 38.1 | 0.4580 | 0.5877 | 0.9009 |
| CUB200 | 40.1 | 0.4740 | 0.9737 | 0.2450 |
| CIFAR-100 | 57.9 | 0.9125 | 1.6429 | 1.3115 |

Acc_T is from CRL model based on Inception-V3 model (Setting-2)

Comparison with ZSL Algorithms

We have evaluated the CRL model in terms of the three perspectives, such as ZSL performances with the six different benchmark datasets, ZSL performance with an increasing number of instances, and comparison with the state-of-the-art ZSL algorithms. The two types of the CRL model (the projection methods) was included Inception-V3 based model and VGG-19 based model. Table 5 and Fig. 7 show the CRL’s ZSL performance with flat-hit@k using Inception-V3 projection with the dataset under Setting-2. Table 6 shows the comparison of the state-of-the-art ZSL algorithms using the VGG-19 model, considering only the IN360 dataset under Setting-2. The experiments show the

CRL model’s performance for the ZSL task that recognizes the target (unseen) labels without having the source (source) labels. Table 5 shows two versions of the recognition tasks with the testing data from the target set (\mathbb{T}); $\mathbb{T} \Rightarrow \mathbb{T}$ when the testing label could be only from the target set $y^* \in \mathbb{T}$ and $\mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$ when the testing label could be from both the source set and target set $y^* \in \mathbb{S} \cup \mathbb{T}$. For this experiment, we consider all instances (70%) from the dataset to generate class representatives. The results show the influence of an increasing number of labels in the dataset to the ZSL accuracy similar observation was made from Fig. 5. When comparing the flat-hit@k with $k = 1$ (Top-1) and $k = 2$ (Top-2), we can see an average of 21% increase from Top-1 accuracy to Top-2 to accuracy. This increase signifies that CRFS provides a well-formed neighborhood, with a 21% increase chance of getting the correct prediction at the second nearest class representative. Comparing the two different recognition tasks, we note the significant drop in efficiency is shown when the source set was also considered. Note that these results were based on Setting-2, source set is ImageNet-1K. The number of classes in $\mathbb{S} \cup \mathbb{T}$ would be a minimum of 1050 (case of AWA2 dataset) and a maximum of 1806 (case of SUN_A dataset). A negative correlation between the number of classes and accuracy was observed.

Figure 7 shows the performance of CR-Inception-V3 with the images at a specified number ranged from one to ten from each class. For this experiment, we considered only ($\mathbb{T} \Rightarrow \mathbb{T}$) setting. Interestingly, the Top-1 accuracy with just ten images reaches higher than 75% of the accuracy compared to the accuracy in Table 5, which considers 70% of the data.

Table 6 shows the state-of-the-art zero-shot learning (ZSL) algorithms to be compared with CRL in Table 1.

Fig. 5 Accuracy distribution of 8 benchmark datasets

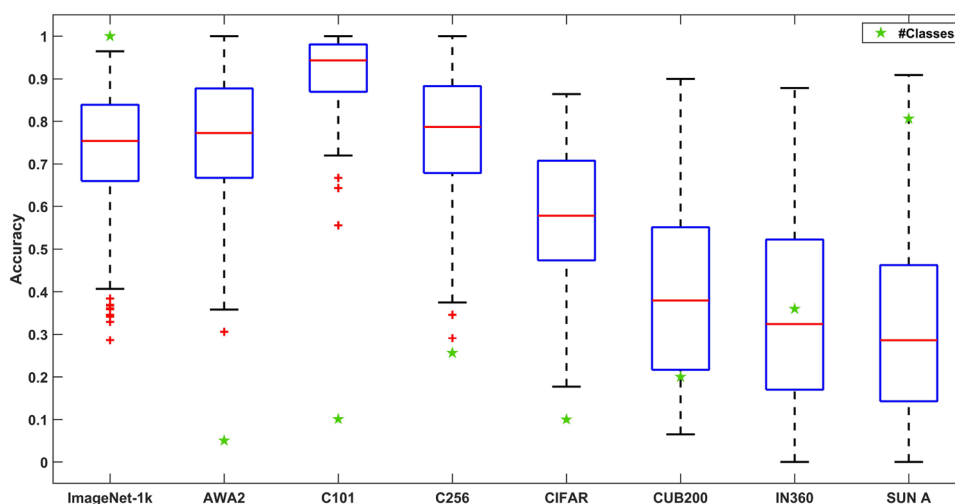


Fig. 6 Inference performance for two best and two worst cases













| SUN_A | | AWA2 | | CUB200 | |
|---|--------------------------------|---|---------------------|---|---------------------------|
|  | Mosque Indoor 16.7% |  | Walrus 19% |  | Cliff Swallow 4.7% |
|  | Lean-to 16.7% |  | Mouse 30% |  | Chipping Sparrow 4.7% |
|  | Aircraft Carrier 100% |  | Giant Panda 100% |  | American Goldfinch 88% |
|  | Badminton Indoor Court 100% |  | Lion 100% |  | Least Auklet 89.6% |

Table 5 Accuracy for zero-shot learning tasks (Setting-2)

| Dataset | Recognition task accuracy | | | Dataset | Recognition task accuracy | | |
|---------|---------------------------|-----------|---------------|---------|---------------------------|-----------|---------------|
| | Flat-hit@K | T ⇒ T (%) | T ⇒ S ∪ T (%) | | Flat-hit@K | T ⇒ T (%) | T ⇒ S ∪ T (%) |
| C-101 | 1 | 91.2 | 86.4 | CUB200 | 1 | 40.1 | 38.4 |
| | 2 | 97.4 | 93.5 | | 2 | 53.9 | 52.5 |
| | 5 | 98.5 | 97.0 | | 5 | 71.0 | 69.9 |
| C-256 | 1 | 70.5 | 55.6 | SUN_A | 1 | 31.9 | 28.5 |
| | 2 | 78.3 | 69.2 | | 2 | 43.3 | 40.4 |
| | 5 | 84.7 | 78.7 | | 5 | 58.7 | 56.3 |
| AWA2 | 1 | 76.8 | 48.6 | 1 | 38.1 | 28.3 | |
| | 2 | 87.9 | 72.9 | 2 | 47.6 | 40.1 | |
| | 5 | 95.5 | 86.5 | 5 | 59.7 | 54.3 | |

Fig. 7 Accuracy on CRL model based on increasing instances in Setting-2 (S ⇒ S and T ⇒ T)

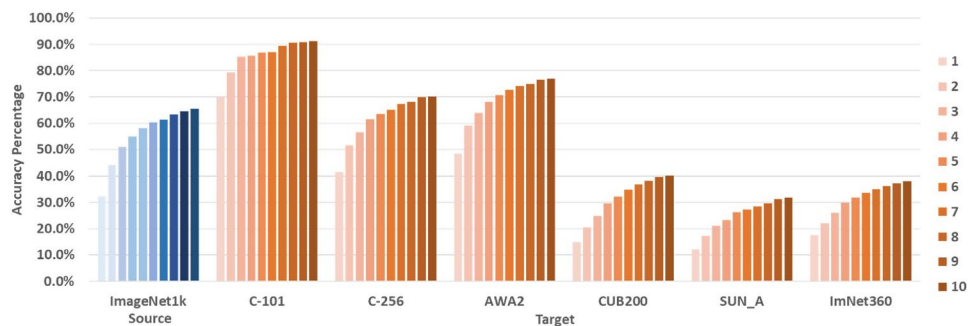


Table 6 Comparison between the CRL model with VGG-19 model for IN360 (Setting-1)

| ZSL type | Methods | 3000 Instances | | All instances | |
|-------------------|--------------|----------------|-----------|---------------|-----------|
| | | Top-1 (%) | Top-5 (%) | Top-1 (%) | Top-5 (%) |
| Projection method | CRL (Ours) | 11.78 | 25.52 | 31.6 | 55.1 |
| | DWV [42, 63] | 9.26 | 21.99 | 10.29 | 23.12 |
| | SAE [45] | 5.11 | 12.26 | 9.32 | 21.04 |
| | Deep-SVR [4] | 5.29 | 13.32 | 5.7 | 14.12 |
| | Embed [46] | - | - | 11.0 | 25.7 |
| | ConSE [6] | 5.5 | 13.1 | 7.8 | 15.5 |
| Correspondence | ESZSL [36] | 5.86 | 13.71 | 8.3 | 18.2 |
| | DeViSE [34] | 3.7 | 11.8 | 5.2 | 12.8 |
| Relationship | AMP [35] | 3.5 | 10.5 | 6.1 | 13.1 |

*Results for SOTA Models are from Fu et al. [42]. The CRL model is configured with the same settings, e.g., VGG-19 with 3000 instances (three images per class) and all 50000 instances, (50 images per class)

This experiment considered the CRL model built using pre-trained VGG-19 as a projection method with ImageNet-1K as a source and IN360 as the target. Table 6 shows that the performance of the CRL model is superior in both cases of 3000 instances and all instances. In the 3000 instance case,

the CRL model’s Top-1 shows a 27% increase compared to the top performer, Deep WMM-Voc. In all cases, the CRL model’s Top-1 accuracy is significantly higher than the others; on average, the CRL model’s Top-1 accuracy is considerably higher than Deep WMM-Voc’s.

Table 7 Accuracy of generalized zero-shot learning algorithms with Setting-1

| Model | AWA2 (Top-1) | | | CUB200 (Top-1) | | | SUN_A (Top-1) | | | IN360 (Top-5) | | |
|--|--------------------|--------------------|-------------|--------------------|--------------------|-------------|--------------------|--------------------|-------------|--------------------|--------------------|-------------|
| | Acc _S ' | Acc _T ' | HM | Acc _S ' | Acc _T ' | HM | Acc _S ' | Acc _T ' | HM | Acc _S ' | Acc _T ' | HM |
| Instance-based projection method | | | | | | | | | | | | |
| CRL-IN (Ours) | 86.0 | 90.8 | 88.3 | 73.3 | 70.4 | 71.8 | 39.7 | 41.2 | 40.5 | 89.4 | 87.5 | 88.4 |
| CRL-RL (Ours) | 87.4 | 83.5 | 85.4 | 47.5 | 54.7 | 50.8 | 37.8 | 41.7 | 39.6 | 75.5 | 70.9 | 73.1 |
| Embed [46] | 84.7 | 32.8 | 47.3 | 57.9 | 19.6 | 29.3 | 34.3 | 20.5 | 25.7 | 72.2 | 22.6 | 34.4 |
| SAE [45] | 71.3 | 31.5 | 43.7 | 36.1 | 28.0 | 31.5 | 25.0 | 15.8 | 19.4 | 89.4 | 21.8 | 35.1 |
| CMT [44] | 86.9 | 8.4 | 15.3 | 60.1 | 4.7 | 8.7 | 28.0 | 8.7 | 13.3 | 70.2 | 11.3 | 19.5 |
| Instance-based synthesizing method | | | | | | | | | | | | |
| BPL+LR [50] | 69.4 | 60.9 | 64.9 | 71.6 | 42.7 | 53.5 | 36.9 | 39.2 | 38.0 | 95.5 | 26.1 | 41.0 |
| CADA-VAE [48] | 72.8 | 57.3 | 64.1 | 53.5 | 51.6 | 52.5 | 35.7 | 47.2 | 40.7 | 74.6 | 25.2 | 37.7 |
| Cycle-GAN [49] | 64.0 | 56.9 | 60.2 | 61.0 | 45.7 | 52.3 | 33.6 | 49.4 | 40.0 | 81.3 | 24.4 | 37.5 |
| GAN+Softmax [47] | 61.4 | 57.9 | 59.6 | 57.7 | 43.7 | 49.7 | 36.6 | 42.6 | 39.4 | 79.1 | 24.2 | 37.1 |
| GAN+AIE [47] | 57.2 | 47.6 | 52.0 | 59.3 | 40.2 | 47.9 | 31.1 | 41.3 | 35.5 | 78.8 | 23.4 | 36.1 |
| Classifier-based relationship method | | | | | | | | | | | | |
| SynC [52] | 87.3 | 8.9 | 16.2 | 70.9 | 11.5 | 19.8 | 43.3 | 7.9 | 13.4 | 94.3 | 10.7 | 19.2 |
| SSE [51] | 80.5 | 7.0 | 12.9 | 46.9 | 8.5 | 14.4 | 36.4 | 2.1 | 4.0 | 84.8 | 10.8 | 19.2 |
| Classifier-based correspondence method | | | | | | | | | | | | |
| SP-AEN [53] | 90.9 | 23.3 | 37.1 | 70.6 | 34.7 | 46.5 | 38.6 | 24.9 | 30.3 | 84.8 | 20.4 | 32.9 |
| AIE [5] | 76.1 | 16.8 | 27.5 | 62.8 | 23.7 | 34.4 | 33.1 | 21.8 | 26.3 | 72.4 | 22.1 | 33.9 |
| DeViSE [34] | 68.7 | 13.4 | 22.4 | 53.0 | 23.8 | 32.8 | 27.4 | 16.9 | 20.9 | 68.9 | 21.8 | 33.1 |
| SJE [33] | 74.6 | 11.3 | 19.6 | 59.2 | 23.5 | 33.6 | 30.5 | 14.7 | 19.8 | 70.1 | 19.5 | 30.5 |
| LATEM [37] | 71.7 | 7.3 | 13.3 | 57.3 | 15.2 | 24.0 | 28.8 | 14.7 | 19.5 | 77.3 | 18.8 | 30.2 |
| ESZSL [36] | 75.6 | 6.6 | 12.1 | 63.8 | 12.6 | 21.0 | 27.9 | 11.0 | 15.8 | 69.1 | 16.5 | 26.6 |

Setting-1: CRL-IN: Inception V3, CRL-RL: ResNet-101; AWA2, CUB200, SUN_A and LeNet: IN360 Results for SOTA G-ZSL Models are from Guan et al. [50]

Bold values indicate that CRL-Inception-v3 based model (with no learning) performs better for target class accuracy for AWA2, CUB200 and IN360 and for SUN_A performs almost as good as other models

Table 8 Transfer learning performance analysis: CRL vs. Inception-V3 [77] pretrained model: Inception-V3 with ImageNet-1K

| Target | CRL-based ZSL setting ($\mathbb{T} \Rightarrow \mathbb{T}$) (Ours) | | | Inception-V3 [77] | | | |
|-----------|--|-------|----------------------|-------------------|----------|----------------------|------|
| | Step | Time* | Acc _T (%) | Time* | Epoch | Acc _T (%) | |
| C-101 | Projection | 5.35 | 7.2 | 94.4 | 4257.91 | 40 | 88.7 |
| | CR generation | 1.51 | | | | | |
| | Inference | 0.33 | | | | | |
| C-256 | Projection | 10.23 | 13.88 | 78.2 | 14193.96 | 14 | 59.3 |
| | CR generation | 1.9 | | | | | |
| | Inference | 1.75 | | | | | |
| CIFAR-100 | Projection | 13.2 | 15.73 | 57.9 | 26941.85 | 10 | 50.3 |
| | CR generation | 0.93 | | | | | |
| | Inference | 1.6 | | | | | |

* Time is calculated in minutes

Bold values indicate the CRL model takes significantly less time with comparable or even better target accuracy than traditional transfer learning and retraining of Inception-V3 on target classes.

Generalized Zero-Shot Learning

Table 7 shows the performance of the generalized zero-shot learning (G-ZSL) model comparing CRL (built on Inception-v3) to the four state-of-the-art G-ZSL models. The SOTA models for AWA2, CUB200, and SUN_A were built on ResNet-101, and the SOTA model for IN360 was built on Google-LeNet. The number of images for the G-ZSL model datasets is as follows: 600 images per class for AWA2, 50 images per class for CUB200, 20 images per class for SUN_A, and 180 images per class for IN360. The number of images is the same as the setting specified by Guan et al., for fair evaluation [50].

Table 7 also shows the accuracy of the target (unseen) and harmonic mean of the CRL model outperforms the other SOTA models, including the BPL+LR model, which includes synthesized data. On average, the accuracy of CRL was 20% better than the other SOTA models. Compared to a fine-grained dataset, the CRL model performs better in the coarse datasets, such as SUN_A with only 40.5% of the Harmonic Mean Accuracy. The CRL accuracies for the seen classes have been improved with all datasets excluding IN360. For the IN360 dataset, BPL+LR shows the highest accuracy of 95.5%, and CRL shows the second-highest accuracy of 89.4%. The pattern of source accuracy being better than the accuracy can be observed in works, such as the instance-based projection method, the classifier-based correspondence method, and the classifier-based relationship method. Reportedly difference in accuracy between source and target domain is around 60% in those three types except CRL. The CRL model is an instance-based projection method that still reports an accuracy equivalent to the instance-based synthesizing method. In the most cases of the experiments, the CRL model outperforms the Synthesizing Methods. The most significant advantage of CRL is a simple, accuracy, and scalable solution, compared to the

synthesizing methods that are complex and time consuming with generative or auto-encoder models.

CRL Time Performance

The CRL model's performance is evaluated by comparing it with the Inception-V3 pre-trained model retrained with the target domain. The CRL model's performance is calculated according to the projection time ("Projection Function"), CR model generation time (refer to "Class Representative Generation") and CR-based inference time (refer to "Class Representative Inference"). For this experiment, we use Inception-V3 as our pre-trained projection. The class representative for this experiment was built with 70% of the given dataset. Note that since most of the ZSL methods do not report time performance, image classification setting was considered. The CRL-based image classification setting is the same as the CRL-based ZSL setting ($\mathbb{T} \Rightarrow \mathbb{T}$) since no learning happens for either of them. Our previous work [68] reports more details on the comparative evaluation of the CRL model with others. The comparison is with the Inception-V3 pre-trained model from MATLAB (same as CRL's projection method), where the last layer of softmax is retrained with the new datasets.

Table 8 shows the comparison of the CRL model's overall time vs. the time taken for retraining the dataset using the Inception-V3 pre-trained model. Both pre-trained models were run on the same system specification. Pre-training of the Inception-V3 Model was stopped at a reported number of epochs as the time taken was significantly higher than the CRL model's. The CRL model with three datasets (CalTech-101, CalTech-256, and CIFAR-100) have an average of 99% time reduction that is a significantly reduced time compared with that for the original Inception-V3 model. Within the same time window and based on the same pre-trained model, the Inception-V3 model performance has not reached the accuracy published in [84]. The CRL model's

overall time shows genuinely outstanding achievements in the target domains, even if the models were never learned from the target domain.

Discussion

The big question of whether our work can be categorized as 'zero-shot' learning. Some researchers understand zero-shot learning in a sense there is no gradient updates [1, 39], including our work. We define CRL model as zero-shot since neither unseen data, unseen labels nor unseen prototypes are used during training phase. The unseen prototypes are generated post training with zero learning. CRL is similar to the generative model in terms of building a prototype class by class, using a class representative criterion [85, 86]. However, they are different since the CR generation is not based on probabilistic, but based on the generalized mean of aggregated features. For the classification step, a standard projection approach is used to compare the prototypes with each other. Compared to existing ZSL algorithms, CRL is required solely visual data, rather than both visual data and auxiliary information. Still, higher accuracy can be achieved compared to the state-of-the-art research in ZSL and G-ZSL. The CR generation was obtained by extracting the abstraction of each feature's distribution in the class representative feature space CRFS for the target domain. For the purpose, we used a straight-forward aggregation approach. Thus, the class representative learning model might be susceptible to outliers, sample size bias, and hubness. The CRL model was extremely strong at the flat-hit@ k with $k = 2$ and $k = 5$ compared to $k = 1$ (Table 5). This indicates that the high similarity between some CRs might lead to misclassification.

To overcome the limitation of the CR generation, an advanced optical model such as the Fisher vector (FV) and Gaussian mixture (GM) models might be incorporated in the future. We will consider using unsupervised deep learning techniques, such as the auto-encoder, for learning efficient data codings and reducing the CR's feature space to a more optimal representation. We can further extend it to determine the CR vectors' common and unique features and find the weights that maximize the uniqueness between CRs. Currently, CRL is mainly based on the use of visual data only for ZSL. In the future, we will extend the CRL model to handle multi-modal data distributions with text data and image data. The CRL model has a potential extension to have the open set recognition with $\mathbb{T} \gg \mathbb{S}$. Currently, CRL has an openness factor of 0.278 [87].

Conclusion

This paper proposes the class representative learning (CRL) that projects the abstract features extracted from a deep learning environment to the high-dimensional visual space. In the CRL model, class representatives (CRs) are designed to

represent potential features for given data from the abstract embedding space. A projection-based inferencing method is intended to reconcile the dominant difference between the seen classes and unseen classes. The CRL model has three distinct advantages than existing ZSL approaches. (1) There is no dependence among CRs so that they can be built in parallel and used freely, depending upon the context. (2) Unlike other ZSL approaches, the CRs can be generated only using the abstract visual space by eliminating the need for semantic spaces or auxiliary information. (3) The abstract embedding space of the source (seen classes) is solely used to project the instances of the target (unseen classes) without any learning involved. The current research demonstrated the benefit of using the class-based approach with class representatives for ZSL and G-ZSL on eight benchmark datasets. Extensive experimental results confirm that the proposed CRL model significantly outperforms the state-of-the-art methods in both ZSL/G-ZSL. The CRL model is presented herein as an instance-based projection-method zero-shot learning method, but surprisingly this outperforms complex state-of-the-art instance-based synthesizing methods.

Acknowledgements This work was partially supported by NSF CNS #1747751. The authors would like to thank Dr. Ye Wang, Associate Professor, UMKC, for constructive criticism of the manuscript

Declarations

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Wang W, Zheng VW, Yu H, Miao C. A survey of zero-shot learning: Settings, methods, and applications. *ACM Trans Intell Syst Technol.* 2019;10(2):13.
2. Lake B, Salakhutdinov R, Gross J, Tenenbaum J. One shot learning of simple visual concepts. In: *Proceedings of the Annual Meeting of the Cognitive Science Society*; 2011. p. 33.
3. Vinyals O, Blundell C, Lillicrap T, Wierstra D, et al. Matching networks for one shot learning. In: *Advances in neural information processing systems*; 2016. pp. 3630–8.
4. Lampert CH, Nickisch H, Harmeling S. Attribute-based classification for zero-shot visual object categorization. *IEEE Trans Pattern Anal Mach Intell.* 2013;36(3):453–65.
5. Akata Z, Perronnin F, Harchaoui Z, Schmid C. Label-embedding for image classification. *IEEE Trans Pattern Anal Mach Intell.* 2015;38(7):1425–38.
6. Norouzi M, Mikolov T, Bengio S, Singer Y, Shlens J, Frome A, Corrado GS, Dean J. Zero-shot learning by convex combination of semantic embeddings. 2013. [arXiv: 1312.5650](https://arxiv.org/abs/1312.5650).
7. Xian Y, Schiele B, Akata Z. Zero-shot learning-the good, the bad and the ugly. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; 2017. pp. 4582–91.
8. Xian Y, Lampert CH, 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(9):2251–65.

9. Kingma DP, Mohamed S, Rezende DJ, Welling M. Semi-supervised learning with deep generative models. In: *Advances in neural information processing systems*; 2014. pp. 3581–9.
10. Diederik PK, Welling M, et al. Auto-encoding variational bayes. In: *Proceedings of the International Conference on Learning Representations (ICLR)*; 2014.
11. Rezende DJ, Mohamed S, Danihelka I, Gregor K, Wierstra D. One-shot generalization in deep generative models. 2016. [arXiv:1603.05106](https://arxiv.org/abs/1603.05106).
12. Xu X, Tsang IW, Liu C. Complementary attributes: a new clue to zero-shot learning. *IEEE Trans Cybern*. 2019.
13. Akata Z, Malinowski M, Fritz M, Schiele B. Multi-cue zero-shot learning with strong supervision. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; 2016. pp. 59–68.
14. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L. ImageNet large scale visual recognition challenge. *Int J Comput Vision*. 2015;115(3):211–52.
15. Kokkinos I. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2017. pp. 6129–38.
16. Rebuffi S-A, Bilen H, Vedaldi A. Learning multiple visual domains with residual adapters. In: *Advances in neural information processing systems*; 2017. pp. 506–16.
17. Bilen H, Vedaldi A. Universal representations: the missing link between faces, text, planktons, and cat breeds. 2017. [arXiv:1701.07275](https://arxiv.org/abs/1701.07275), 2017.
18. Rebuffi S-A, Bilen H, Vedaldi A. Efficient parametrization of multi-domain deep neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2018. pp. 8119–27.
19. Kapoor A, Lee B, Tan D, Horvitz E. Learning to learn: Algorithmic inspirations from human problem solving. In: *Twenty-Sixth AAAI Conference on Artificial Intelligence*; 2012.
20. Shen L, Sun G, Huang Q, Wang S, Lin Z, Wu E. Multi-level discriminative dictionary learning with application to large scale image classification. *IEEE Trans Image Process*. 2015;24(10):3109–23.
21. Changpinyo S, Chao W-L, Sha F. Predicting visual exemplars of unseen classes for zero-shot learning. In: *Proceedings of the IEEE international conference on computer vision*; 2017. pp. 3476–85.
22. Zhang X, Gui S, Zhu Z, Zhao Y, Liu J. Hierarchical prototype learning for zero-shot recognition. *IEEE Trans Multimedia*. 2019.
23. Fu Z, Xiang T, Kodirov E, Gong S. Zero-shot learning on semantic class prototype graph. *IEEE Trans Pattern Anal Mach Intell*. 2017;40(8):2009–22.
24. Snell J, Swersky K, Zemel R. Prototypical networks for few-shot learning. In: *Advances in neural information processing systems*; 2017. pp. 4077–87.
25. Palatucci M, Pomerleau D, Hinton GE, Mitchell, TM. Zero-shot learning with semantic output codes. In: *Advances in neural information processing systems*; 2009. pp. 1410–18.
26. Wolf L, Martin, I. Robust boosting for learning from few examples. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. IEEE; 2005. pp. 1:359–64.
27. Torralba A, Murphy KP, Freeman WT. Sharing visual features for multiclass and multiview object detection. *IEEE Trans Pattern Anal Mach Intell*. 2007;5:854–69.
28. Fleuret F, Blanchard, G. Pattern recognition from one example by chopping. In: *Advances in neural information processing systems*; 2006. pp. 371–78.
29. Bart E, Ullman S. Cross-generalization: Learning novel classes from a single example by feature replacement. In: *IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. IEEE;2005. pp. 672–9.
30. Amit Y, Fink M, Srebro N, Ullman, S. Uncovering shared structures in multiclass classification. In: *Proceedings of the 24th international conference on Machine learning*. ACM; 2007. pp. 17–24.
31. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*; 2013. pp. 3111–19.
32. Pennington J, Socher R, Manning C. Glove: global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014. pp. 1532–43.
33. Akata Z, Reed S, Walter D, Lee H, Schiele B. Evaluation of output embeddings for fine-grained image classification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; 2015. pp. 2927–36.
34. Frome A, Corrado GS, Shlens J, Bengio S, Dean J, Ranzato M, Mikolov T. Devise: a deep visual-semantic embedding model. In: *Advances in neural information processing systems*; 2013. pp. 2121–9.
35. Fu Z, Xiang T, Kodirov E, Gong S. Zero-shot object recognition by semantic manifold distance. In: *The IEEE conference on computer vision and pattern recognition (CVPR)*. 2015.
36. Romera-Paredes B, Torr P. An embarrassingly simple approach to zero-shot learning. In: *International conference on machine learning*; 2015. pp. 2152–61.
37. Xian Y, Akata Z, Sharma G, Nguyen Q, Hein M, Schiele B. Latent embeddings for zero-shot classification. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2016. pp. 69–77.
38. Yu Y, Zhang Z, Han J. Meta-transfer networks for zero-shot learning. 2019. [arXiv:1909.03360](https://arxiv.org/abs/1909.03360).
39. Wang Q, Chen K. Alternative semantic representations for zero-shot human action recognition. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer; 2017. pp. 87–102.
40. Wang Q, Bu P, Breckon TP. Unifying unsupervised domain adaptation and zero-shot visual recognition. 2019. [arXiv:1903.10601](https://arxiv.org/abs/1903.10601).
41. Zhu Y, Long Y, Guan Y, Newsam S, Shao L. Towards universal representation for unseen action recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2018. pp. 9436–45.
42. Fu Y, Wang X, Dong H, Jiang Y-G, Wang M, Xue X, Sigal L. Vocabulary-informed zero-shot and open-set learning. *IEEE Trans Pattern Anal Mach Intell*. 2019.
43. Farhadi A, Endres I, Hoiem D, Forsyth D, Describing objects by their attributes. In: *IEEE conference on computer vision and pattern recognition*. IEEE; 2009. pp. 1778–85.
44. Socher R, Ganjoo M, Manning CD, Ng A. Zero-shot learning through cross-modal transfer. In: *Advances in neural information processing systems*. 2013; pp. 935–43.
45. Kodirov E, Xiang T, Gong S. Semantic autoencoder for zero-shot learning. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2017. pp. 3174–83.
46. Zhang L, Xiang T, Gong S. Learning a deep embedding model for zero-shot learning. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2017. pp. 2021–30.
47. Xian Y, Lorenz T, Schiele B, Akata Z. Feature generating networks for zero-shot learning. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2018. pp. 5542–51.
48. 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 conference on computer vision and pattern recognition*; 2019. pp. 8247–55.

49. Felix R, Kumar VB, Reid I, Carneiro G. Multi-modal cycle-consistent generalized zero-shot learning. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. pp. 21–37.
50. Guan J, Lu Z, Xiang T, Li A, Zhao A, Wen J-R. Zero and few shot learning with semantic feature synthesis and competitive learning. *IEEE Trans Pattern Anal Mach Intell.* 2020.
51. Zhang Z, Saligrama V. Zero-shot learning via semantic similarity embedding. In: Proceedings of the IEEE international conference on computer vision; 2015. pp. 4166–74.
52. Changpinyo S, Chao W-L, Gong B, Sha F. Synthesized classifiers for zero-shot learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. pp. 5327–36.
53. Chen L, Zhang H, Xiao J, Liu W, Chang S-F. Zero-shot visual recognition using semantics-preserving adversarial embedding networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. pp. 1043–52.
54. Akata Z, Perronnin F, Harchaoui Z, Schmid C. Label-embedding for attribute-based classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2013. pp. 819–26.
55. Lei Ba J, Swersky K, Fidler S, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In: Proceedings of the IEEE international conference on computer vision; 2015. pp. 4247–55.
56. Yazdani M, Henderson J. A model of zero-shot learning of spoken language understanding. In: Proceedings of the 2015 conference on empirical methods in natural language processing; 2015. pp. 244–9.
57. Mikolov T, Le QV, Sutskever I. Exploiting similarities among languages for machine translation. 2013. [arXiv:1309.4168](https://arxiv.org/abs/1309.4168).
58. Shigeto Y, Suzuki I, Hara K, Shimbo M, Matsumoto Y. “Ridge regression, hubness, and zero-shot learning. In: Joint European Conference on machine learning and knowledge discovery in databases. Springer; 2015. pp. 135–51.
59. Long Y, Liu L, Shen F, Shao L, Li X. Zero-shot learning using synthesised unseen visual data with diffusion regularisation. *IEEE Trans Pattern Anal Mach Intell.* 2017;40(10):2498–512.
60. Xu X, Hospedales T, Gong S. Transductive zero-shot action recognition by word-vector embedding. *Int J Comput Vision.* 2017;123(3):309–33.
61. Kodirov E, Xiang T, Fu Z, Gong S. Unsupervised domain adaptation for zero-shot learning. In: Proceedings of the IEEE international conference on computer vision; 2015. pp. 2452–60.
62. Fu Y, Hospedales TM, Xiang T, Gong S. Learning multi-modal latent attributes. *IEEE Trans Pattern Anal Mach Intell.* 2013;36(2):303–16.
63. Fu Y, Sigal L. Semi-supervised vocabulary-informed learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. pp. 5337–46.
64. Wang Q, Chen K. Zero-shot visual recognition via bidirectional latent embedding. *Int J Comput Vision.* 2017;124(3):356–83.
65. Xu B, Fu Y, Jiang Y-G, Li B, Sigal L. Video emotion recognition with transferred deep feature encodings. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval; 2016. pp. 15–22.
66. Xu X, Shen F, Yang Y, Zhang D, Tao Shen H, Song J. Matrix trifactorization with manifold regularizations for zero-shot learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. pp. 3798–07.
67. Tamaazousti Y, Le Borgne H, Hudelot C, Seddik MEA, Tamaazousti M. Learning more universal representations for transfer-learning. *IEEE Trans Pattern Anal Mach Intell.* 2019.
68. Chandrashekar M, Lee Y. Crl: Class representative learning for image classification. 2020. [arXiv:2002.06619](https://arxiv.org/abs/2002.06619).
69. Raina R, Battle A, Lee H, Packer B, Ng AY. Self-taught learning: transfer learning from unlabeled data. In: Proceedings of the 24th international conference on machine learning. ACM; 2007. pp. 759–66.
70. Ben-David S, Blitzer J, Crammer K, Pereira F. Analysis of representations for domain adaptation. In: Advances in neural information processing systems; 2007. pp. 137–44.
71. Blitzer J, Crammer K, Kulesza A, Pereira F, Wortman J. Learning bounds for domain adaptation. In: Advances in neural information processing systems; 2008. pp. 129–36.
72. Kifer D, Ben-David S, Gehrke J. Detecting change in data streams. In: VLDB, vol. 4. Toronto; 2004. pp. 180–91.
73. Fei-Fei L, Fergus R, Perona P. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In: Conference on computer vision and pattern recognition workshop. IEEE; 2004. p. 178.
74. Griffin G, Holub A, Perona P. Caltech-256 object category dataset. 2007.
75. Welinder P, Branson S, Mita T, Wah C, Schroff F, Belongie S, Perona P. Caltech-UCSD Birds 200. Tech. Rep. CNS-TR-2010-001, California Institute of Technology. 2010.
76. Patterson G, Xu C, Su H, Hays J. The sun attribute database: beyond categories for deeper scene understanding. *Int J Comput Vision.* 2014;108(1–2):59–81.
77. Pretrained inception-v3 convolutional neural network - matlab inceptionv3 - mathworks india. 2019. <https://in.mathworks.com/help/deeplearning/ref/inceptionv3.html>. (Accessed on 05/13/2019).
78. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. pp. 2818–26.
79. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. pp. 770–778.
80. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014. [arXiv:1409.1556](https://arxiv.org/abs/1409.1556).
81. 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; 2015. pp. 1–9.
82. Foundation AS “Apache spark”. 2019. Accessed on 26 May 2019.
83. Day O, Khoshgoftaar TM. A survey on heterogeneous transfer learning. *J Big Data.* 2017;4(1):29.
84. Kornblith S, Shlens J, Le QV. Do better imagenet models transfer better? 2018. [arXiv:1805.08974](https://arxiv.org/abs/1805.08974).
85. Ulusoy I, Bishop CM. Comparison of generative and discriminative techniques for object detection and classification. In: Toward category-level object recognition. Springer; 2006. pp. 173–95.
86. Wang W, Pu Y, Verma VK, Fan K, Zhang Y, Chen C, Rai P, Carin L. Zero-shot learning via class-conditioned deep generative models. In: Thirty-second AAAI conference on artificial intelligence. 2018.
87. Scheirer WJ, de Rezende Rocha A, Sapkota A, Boult TE. Toward open set recognition. *IEEE Trans Pattern Anal Mach Intell.* 2012;35:1757–72.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.