A Survey of Zero-Shot Learning: Settings, Methods, and Applications

WEI WANG, Nanyang Technological University, Singapore
VINCENT W. ZHENG, WeBank, China
HAN YU, Nanyang Technological University, Singapore
CHUNYAN MIAO, Nanyang Technological University, Singapore

Most machine-learning methods focus on classifying instances whose classes have already been seen in training. In practice, many applications require classifying instances whose classes have not been seen previously. Zero-shot learning is a powerful and promising learning paradigm, in which the classes covered by training instances and the classes we aim to classify are disjoint. In this paper, we provide a comprehensive survey of zero-shot learning. First of all, we provide an overview of zero-shot learning. According to the data utilized in model optimization, we classify zero-shot learning into three learning settings. Second, we describe different semantic spaces adopted in existing zero-shot learning works. Third, we categorize existing zero-shot learning methods and introduce representative methods under each category. Fourth, we discuss different applications of zero-shot learning. Finally, we highlight promising future research directions of zero-shot learning.

CCS Concepts: • **Computing methodologies** → **Transfer learning**;

Additional Key Words and Phrases: Zero-Shot Learning Survey

ACM Reference Format:
Wei Wang, Vincent W. Zheng, Han Yu, and Chunyan Miao. 2019. A Survey of Zero-Shot Learning: Settings, Methods, and Applications. *ACM Trans. Intell. Syst. Technol.* 10, 2, Article 13 (January 2019), 37 pages. https://doi.org/10.1145/3293318

1 INTRODUCTION

Supervised classification methods have achieved significant success in research and have been applied in many areas. Especially in recent years, benefiting from the fast development of deep learning techniques, they have made much progress. However, there are some restrictions for methods under this learning paradigm. In supervised classification, sufficient labeled training instances are needed for each class. In addition, the learned classifier can only classify the instances belonging to classes covered by the training data, and lacks the ability to deal with previously unseen classes. However, in practical applications, there may not be sufficient training instances for

Authors' addresses: Wei Wang, Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), Interdisciplinary Graduate School, and School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore, wwang008@e.ntu.edu.sg; Vincent W. Zheng, WeBank, Shahexilu 1819, Nanshan, Shenzhen, China, wenchenzheng@gmail.com; Han Yu, School of Computer Science and Engineering, and Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), Nanyang Technological University, 50 Nanyang Avenue, Singapore, han.yu@ntu.edu.sg; Chunyan Miao, School of Computer Science and Engineering, and Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), Nanyang Technological University, 50 Nanyang Avenue, Singapore, ascymiao@ntu.edu.sg.

2157-6904/2019/01-ART13 $15.00
https://doi.org/10.1145/3293318

each class. There could also be situations in which the classes not covered by the training instances appearing in the testing instances.

To deal with these problems, methods under different learning paradigms have been proposed. To tackle the problem of learning classifiers for classes with few training instances, *few-shot learning / one-shot learning* methods [35, 122] have been proposed. In these methods, while learning classifiers for the classes with few instances, knowledge contained in instances of other classes is utilized. To deal with previously unseen classes, a range of learning methods have been proposed. In *open set recognition* methods [64, 132], when learning the classifier with the training data, the fact of unseen classes existing is taken in consideration. The learned classifier can determine whether a testing instance belongs to the unseen classes, but it cannot determine which specific unseen class the instance belongs to. The *cumulative learning* [34] and *class-incremental learning* [123] methods have been proposed for problems in which labeled instances belonging to some previously unseen classes progressively appear after model learning. The learned classifier can be adapted with these newly available labeled instances to be able to classify classes covered by them. The *open world recognition* [9] methods follow the process of "unseen classes detection, labeled instances of unseen classes acquisition, and model adaption", and adapt the classifier to be able to classify previously unseen classes with the acquired labeled instances belonging to them.

For methods under the above learning paradigms, if the testing instances belong to unseen classes that have no available labeled instances during model learning (or adaption), the learned classifier cannot determine the class labels of them. However, in many practical applications, we need the classifier to have the ability to determine the class labels for the instances belonging to these classes. The following are some popular application scenarios:

- *The number of target classes is large.* An example is object recognition in computer vision. Generally, human beings can recognize at least 30,000 object classes [10]. However, collecting sufficient labeled instances for such a large number of classes is challenging. Thus, existing image datasets can only cover a small subset of these classes. There are many object classes having no labeled instances in existing datasets. A similar example is activity recognition [178], where the number of human activities is large, in contrast with the limited number of activity classes covered by existing datasets. Many activity classes have no labeled instances in existing datasets.
- *Target classes are rare.* An example is fine-grained object classification. Suppose we want to recognize flowers of different breeds [8, 31]. It is hard to collect sufficient image instances for each specific flower breed. For many rare breeds, we cannot find the corresponding labeled instances.
- *Target classes change over time.* An example is recognizing images of products belonging to certain style and brand. As products of new styles and new brands appear frequently, for some new products, it is difficult to find corresponding labeled instances [98].
- *In some particular tasks, it is expensive to obtain labeled instances.* In some learning tasks related with classification, the instance labeling process is expensive and time consuming. Thus, the number of classes covered by existing datasets is limited, and many classes have no labeled instances. For example, in the image semantic segmentation problem [93, 104], the images used as training data should be labeled at the pixel level. This problem can be seen as a pixel-level classification problem for the images. The number of object classes covered by existing datasets is limited, with many object classes having no labeled instances. In another example, in the image captioning problem [140], each image in the training data should have a corresponding caption. This problem can be seen as a sequential classification problem. The

number of object classes covered by the existing image-text corpora is limited, with many object classes not being covered.

In these applications, there are many classes having no labeled instances. For a classifier, it is important for it to have the ability to determine the class label of instances belonging to these classes. To solve this problem, *zero-shot learning* (also known as *zero-data learning* [81]) is proposed. The aim of zero-shot learning is to classify instances belonging to the classes that have no labeled instances. Since its inception [80, 81, 113], zero-shot learning has become a fast-developing field in machine learning, with a wide range of applications in computer vision, natural language processing, and ubiquitous computing.

1.1 Overview of Zero-Shot Learning

In zero-shot learning, there are some labeled training instances in the feature space. The classes covered by these training instances are referred to as the *seen classes*. In the feature space, there are also some unlabeled testing instances, which belong to another set of classes. These classes are referred to as the *unseen classes*. The feature space is usually a real number space and each instance is represented as a vector within it. Each instance is usually assumed to belong to one class[1]. Now, we give the definition of zero-shot learning. Denote $\mathcal{S} = \{c_i^s | i = 1, \ldots, N_s\}$ as the set of seen classes, where each c_i^s is a seen class. Denote $\mathcal{U} = \{c_i^u | i = 1, \ldots, N_u\}$ as the set of unseen classes, where each c_i^u is an unseen class. Note that $\mathcal{S} \cap \mathcal{U} = \emptyset$. Denote $\mathcal{X}$ as the feature space, which is D-dimensional, usually it is a real number space $\mathbb{R}^D$. Denote $D^{tr} = \{(\mathbf{x}_i^{tr}, y_i^{tr}) \in \mathcal{X} \times \mathcal{S}\}_{i=1}^{N_{tr}}$ as the set of labeled training instances belonging to seen classes; for each labeled instance $(\mathbf{x}_i^{tr}, y_i^{tr})$, $\mathbf{x}_i^{tr}$ is the instance in the feature space, and y_i^{tr} is the corresponding class label. Denote $X^{te} = \{\mathbf{x}_i^{te} \in \mathcal{X}\}_{i=1}^{N_{te}}$ as the set of testing instances, where each $\mathbf{x}_i^{te}$ is a testing instance in the feature space. Denote $Y^{te} = \{y_i^{te} \in \mathcal{U}\}_{i=1}^{N_{te}}$ as the corresponding class labels for X^{te}, which are to be predicted.

Definition 1.1 (Zero-Shot Learning). Given labeled training instances D^{tr} belonging to the seen classes $\mathcal{S}$, zero-shot learning aims to learn a classifier $f^u(\cdot) : \mathcal{X} \rightarrow \mathcal{U}$ that can classify testing instances X^{te} (i.e., to predict Y^{te}) belonging to the unseen classes $\mathcal{U}$.

From the definition, we can see that the general idea of zero-shot learning is to transfer the knowledge contained in the training instances D^{tr} to the task of testing instance classification. The label spaces covered by the training and the testing instances are disjoint. Thus, zero-shot learning is a subfield of *transfer learning* [114, 115]. In transfer learning, knowledge contained in the source domain and source task is transferred to the target domain for learning the model in the target task [114, 115]. According to [23, 114], based on whether the feature spaces and label spaces in the source and target domains/tasks are the same, transfer learning can be classified into *homogeneous transfer learning* and *heterogeneous transfer learning*. In homogeneous transfer learning, the feature spaces and the label spaces are the same; while in heterogeneous transfer learning, the feature spaces and/or the label spaces are different. In zero-shot learning, the source feature space is the feature space of training instances, and the target feature space is the feature space of testing instances. They are the same, both are $\mathcal{X}$. However, the source label space is the seen class set $\mathcal{S}$, while the target label space is the unseen class set $\mathcal{U}$. They are different. In view of this, zero-shot learning belongs to heterogeneous transfer learning. Specifically, it belongs to heterogeneous transfer learning with different label spaces (we briefly refer to it as HTL-DLS). Many existing methods for HTL-DLS are proposed for problems under the setting in which there are some labeled instances for the target label space classes [23]. However, in zero-shot learning,

[1]There are some works on zero-shot learning under the multi-label setting, which focuses on classifying instances that each has more than one class label. We will separately discuss these works in Section 3.3.

no labeled instances belonging to the target label space classes (the unseen classes) are available. This makes the problems in zero-shot learning different from these problems studied in HTL-DLS.

Auxiliary information. As no labeled instances belonging to the unseen classes are available, to solve the zero-shot learning problem, some *auxiliary information* is necessary. Such auxiliary information should contain information about all of the unseen classes. This is to guarantee that each of the unseen classes is provided with corresponding auxiliary information. Meanwhile, the auxiliary information should be related to the instances in the feature space. This is to guarantee the auxiliary information is usable.

In existing works, the approach to involve auxiliary information is inspired by the way human beings recognize the world. Humans can perform zero-shot learning with the help of some semantic background knowledge. For example, with the knowledge that "a zebra looks like a horse, and with stripes", we can recognize a zebra even without having seen one before, as long as we know what a horse looks like and what the pattern "stripe" looks like [45]. In this way, the auxiliary information involved by existing zero-shot learning methods is usually some *semantic information*. It forms a space which contains both the seen and the unseen classes. As this space contains semantic information, it is often referred to as the *semantic space*. Being similar to the feature space, the semantic space is also usually a real number space. In the semantic space, each class has a corresponding vector representation, which is referred to as the *class prototype* (or *prototype* for short) of this class[2]. We denote $\mathcal{T}$ as the semantic space. Suppose $\mathcal{T}$ is M-dimensional; it is usually $\mathbb{R}^M$. Denote $\mathbf{t}_i^s \in \mathcal{T}$ as the class prototype for seen class c_i^s, and $\mathbf{t}_i^u \in \mathcal{T}$ as the class prototype for unseen class c_i^u. Denote $T^s = \{\mathbf{t}_i^s\}_{i=1}^{N_s}$ as the set of prototypes for seen classes, and $T^u = \{\mathbf{t}_i^u\}_{i=1}^{N_u}$ as the set of prototypes for unseen classes. Denote $\pi(\cdot) : \mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{T}$ as a class prototyping function that takes a class label as input and outputs the corresponding class prototype. In zero-shot learning, along with the training instances D^{tr}, the class prototypes T^s and T^u are also involved in obtaining the zero-shot classifier $f^u(\cdot)$. In Section 2, we will categorize and introduce different kinds of semantic spaces.

We summarize the key notations used throughout this paper in Table 1.

1.2 Learning Settings

In zero-shot learning, the goal is to learn the zero-shot classifier $f^u(\cdot)$. During model learning, if information about the testing instances is involved, the learned model is transductive for these specific testing instances. In zero-shot learning, this transduction can be embodied in two progressive degrees: transductive for specific unseen classes and transductive for specific testing instances. This is different from the well known transductive setting in semi-supervised learning, which is just for the testing instances. In the setting that is transductive for specific unseen classes, information about the unseen classes is involved in model learning, and the model is optimized for these specific unseen classes. In the setting that is transductive for specific testing instances, the transductive degree goes further. The testing instances are also involved in model learning, and the model is optimized for these specific testing instances. Based on the degree of transduction, we categorize zero-shot learning into three learning settings.

Definition 1.2 (Class-Inductive Instance-Inductive (CIII) Setting). Only labeled training instances D^{tr} and seen class prototypes T^s are used in model learning.

Definition 1.3 (Class-Transductive Instance-Inductive (CTII) Setting). Labeled training instances D^{tr}, seen class prototypes T^s and unseen class prototypes T^u are used in model learning.

[2]In some works, for each class, there is more than one corresponding prototype in the semantic space. We will separately discuss them in Section 3.3.

Table 1. Key Notations Used in This Article

Notation	Description
$\mathcal{X}$	Feature space, which is D-dimensional
$\mathcal{T}$	Semantic space, which is M-dimensional
$\mathcal{S}, \mathcal{U}$	Set of seen classes and set of unseen classes, respectively
N_{tr}, N_{te}	Number of training instances and number of testing instances, respectively
N_s, N_u	Number of seen classes and number of unseen classes, respectively
D^{tr}	The set of labeled training data from seen classes
X^{te}	The set of testing instances from unseen classes
Y^{te}	Labels for testing instances
$(\mathbf{x}_i^{tr}, y_i^{tr})$	The i-th labeled training instance: features $\mathbf{x}_i^{tr} \in \mathcal{X}$ and label $y_i^{tr} \in \mathcal{S}$
$\mathbf{x}_i^{te}$	The i-th unlabeled testing instance: features $\mathbf{x}_i^{te} \in \mathcal{X}$
T^s, T^u	The set of prototypes for seen classes and unseen classes, respectively
$(c_i^s, \mathbf{t}_i^s)$	The i-th seen class $c_i^s \in \mathcal{S}$ and its class prototype $\mathbf{t}_i^s \in \mathcal{T}$
$(c_i^u, \mathbf{t}_i^u)$	The i-th unseen class $c_i^u \in \mathcal{U}$ and its class prototype $\mathbf{t}_i^u \in \mathcal{T}$
$\pi(\cdot)$	A class prototyping function $\pi(\cdot) : \mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{T}$
$f^u(\cdot)$	A zero-shot classifier $f^u(\cdot) : \mathcal{X} \rightarrow \mathcal{U}$

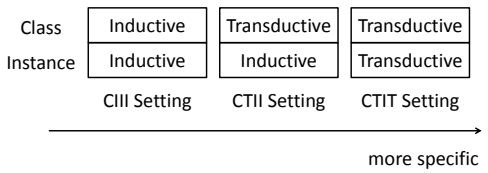
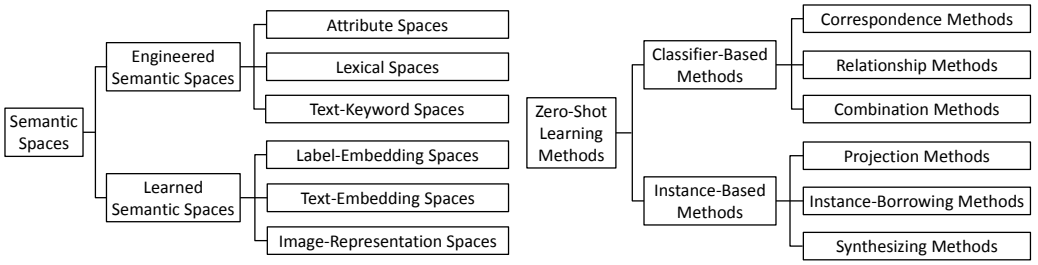


Fig. 1. Different learning settings for zero-shot learning.

Definition 1.4 (Class-Transductive Instance-Transductive (CTIT) Setting). Labeled training instances D^{tr}, seen class prototypes T^s, unlabeled testing instances X^{te} and unseen class prototypes T^u are used in model learning.

We summarize the above three learning settings in Figure 1. As we can see, from CIII to CTIT, the classifier $f^u(\cdot)$ is learned with increasingly specific testing instances' information. In machine learning methods, as the distributions of the training and the testing instances are different, the performance of the model learned with the training instances will decrease when applied to the testing instances. This phenomenon is more severe in zero-shot learning, as the classes covered by the training and the testing instances are disjoint [6, 41]. In zero-shot learning, this phenomenon is usually referred to as *domain shift* [41].

Under the CIII setting, as no information about the testing instances is involved in model learning, the problem of domain shift is severe in some methods under this setting. However, as the models under this setting are not optimized for specific unseen classes and testing instances, when new unseen classes or testing instances need to be classified, the generalization ability of models learned under this setting is usually better than models learned under the CTII or CTIT settings. Under the CTII setting, as the unseen class prototypes are involved in model learning, the problem of domain shift is less severe. However, the ability of CTII methods to generalize to new unseen classes is limited. Further, under the CTIT setting, as the models are optimized for specific unseen classes and testing instances, the problem of domain shift is the least severe among these three learning settings. However, the generalization ability to new unseen classes and testing instances is also the



(a) Different semantic spaces. (b) Different categories of methods.

Fig. 2. Different semantic spaces and methods in zero-shot learning.

most limited. In Section 3, when introducing existing methods in each category, we will introduce methods under these three learning settings separately.

1.3 Contributions and Article Organization

Our contributions. As mentioned above, zero-shot learning is a subfield of transfer learning, belonging to heterogeneous transfer learning. Although there are surveys on transfer learning [115, 152], and particularly on heterogeneous transfer learning [23], they do not cover the topic of zero-shot learning with sufficient depth. To the best of our knowledge, only a few attempts have been made for literature review on zero-shot learning. In [136], about 30 works on zero-shot learning are reviewed. Based on how the feature space and the semantic space are related, this article categorizes the zero-shot learning methods into three categories: one-order transformation approaches, two-order transformation approaches, and high-order transformation approaches. However, as the number of related works reviewed by this article is limited, this categorization is also limited. Many existing methods do not belong to any of these three categories. On the other hand, in this article, just a brief introduction of some semantic spaces is given. No formal categorization of existing semantic spaces is provided. In [155], 16 methods in zero-shot learning are reviewed. A categorization of these 16 methods is given. However, in this article, the emphasis is not the summarization of existing works, rather it is the evaluation of existing zero-shot learning methods. Thus, they give detailed descriptions of the proposed experimental design and evaluation protocols. The performance of the 16 methods under these protocols is also discussed. However, the criteria of method categorization are not given. In addition, no summarization of the semantic spaces is given. In [43], they use a unified "embedding model and recognition in embedding space" framework to summarize the existing methods in zero-shot learning. In the part about zero-shot learning methods, about 40 papers are reviewed. However, the containment of this framework is limited, and many existing methods are not within this framework. On the other hand, in [43], the semantic spaces are categorized based on whether they are attribute spaces or not (attribute spaces are a kind of semantic spaces as shown in Figure 2a). Under this categorization, although the attribute spaces can be introduced in detail, the introduction of other semantic spaces is not sufficient. Given the fast development and increasingly wide application of zero-shot learning, a comprehensive survey of zero-shot learning, which covers a systematic categorization of learning settings, methods, semantic spaces and applications is needed.

We summarize our contributions in this survey paper as follows: (1) As shown in Figure 2b, we provide a hierarchical categorization of existing methods in zero-shot learning. We first categorize methods into two general categories based on they aim to get the classifiers for the unseen classes

directly or aim to get the instances of the unseen classes. Then, methods in each general category are further categorized. Compared with [43, 136, 155], our categorization provides a more comprehensive perspective for readers to understand the existing zero-shot learning methods and select suitable ones for the application scenarios they encounter. (2) We provide a formal classification and definition of different learning settings in zero-shot learning. To the best of our knowledge, we are the first to perform this work. (3) As shown in Figure 2a, we provide a categorization of existing semantic spaces in zero-shot learning. Different from [43] which is based on whether the semantic spaces are attribute spaces, we categorize existing semantic spaces based on how they are constructed.

Article organization. In Section 2, we introduce different kinds of semantic spaces that are used in existing zero-shot learning works. In Section 3, we categorize existing zero-shot learning methods, then we introduce methods in each category, and give a discussion of different methods in the end. In Section 4, we present several typical applications of zero-shot learning. In Section 5, we suggest several future research directions for zero-shot learning. Finally, we conclude this survey article in Section 6.

2 SEMANTIC SPACES

Semantic spaces contain semantic information about classes, and is an important part of zero-shot learning. Various semantic spaces have been exploited by existing works. According to how a semantic space is constructed, we categorize them as (1) engineered semantic spaces, and (2) learned semantic spaces. A detailed taxonomy of the semantic spaces is shown in Figure 2a.

2.1 Engineered Semantic Spaces

In engineered semantic spaces, each dimension of the semantic space is designed by humans. There are different kinds of engineered semantic spaces, each of which has a unique data source and a unique way to construct the space. Next, we introduce typical engineered semantic spaces that are used in zero-shot learning.

Attribute spaces. Attribute spaces are kinds of semantic spaces that are constructed by a set of *attributes*. They are one of the most widely used semantic spaces in zero-shot learning. In the pioneering works [80, 113], attribute spaces have already been adopted. In an attribute space, a list of terms describing various properties of the classes are defined as attributes. Each attribute is usually a word or a phrase, corresponding to one property of these classes. For example, in the problem of animal recognition in images, the attributes could be body color (e.g., "gray", "brown" and "yellow"), or habitat (e.g., "coastal", "desert" and "forest") [80]. Then, these attributes are used to form the semantic space, with each dimension being one attribute. For each class, the values of each dimension of the corresponding prototype is determined by whether this class has the corresponding attribute. Suppose in the animal recognition problem, there are three attributes: "having stripes", "living on land" and "plant eating". For the class "tiger", the corresponding values of these three attributes form the prototype $[1, 1, 0]$. For the class "horse", the corresponding values form the prototype $[0, 1, 1]$ [80]. In the above example, the attribute values are binary (i.e., $0/1$). Thus, the resulting attribute space is referred to as a *binary attribute space*. In general, the attribute values can also be real numbers, indicating the degree/confidence-level for one class having an attribute. The resulting attribute space is referred to as a *continuous attribute space*. Binary and continuous attribute spaces are the most common attribute spaces. In addition, there also exist *relative attribute spaces* [116], which measure the relative degree of having an attribute among different classes.

Lexical spaces. Lexical spaces are kinds of semantic spaces that are constructed by a set of *lexical items*. Lexical spaces are based on the labels of the classes and datasets that can provide semantic information. The datasets can be some structured lexical databases, such as WordNet. Adopting WordNet as information source, there are different ways to construct semantic spaces. One way is by leveraging the hierarchical relationships in WordNet. Such as using all seen and unseen classes, together with their ancestors in WordNet to form the semantic space, with each dimension corresponding to one class [3, 4]. Then, for a class c_i, the value of the jth dimension of its prototype $\mathbf{t}_i$ is determined by the relation between class c_i and class c_j (the corresponding class of the jth dimension). In some approaches, the value is 1 if c_j is the ancestor of c_i or is c_i itself; otherwise, the value is 0 [3]. In other approaches, the value is determined by the distance between c_i and c_j in WordNet; the distance measure can be Jiang-Conrath distance [4], Lin distance [4] or path length similarity [4]. Besides hierarchical relation, other relationships such as partial relationships in WordNet can also be used to form the semantic space [130]. WordNet is the representation of general lexical databases. In some problems, there are problem-specific lexical databases. For example, in fine-grained named entity typing [99], there are predefined tree hierarchies of different entity types, which are used to form the semantic space. Along with the structured lexical databases, the datasets could also be some corpora. For example, in [113], each class is represented as a co-occurrence vector of the class label with the 5,000 most frequent words from the Google Trillion-Word-Corpus.

Text-keyword spaces. Text-keyword spaces are kinds of semantic spaces that are constructed by a set of keywords extracted from the text descriptions of each class. In text-keyword spaces, the most common source of the text descriptions is websites, including both general websites like Wikipedia [4, 120], and domain specific websites. For example, in [8, 31], as the task is zero-shot flower recognition in images, both the Plant Database and Plant Encyclopedia are used (which are specific for plants) to obtain the text descriptions for each flower class. In addition to predefined websites, such text descriptions can also be obtained from search engines. For example, in [178], each class name is used as a query to Google to find the web pages describing the class. In some specific problems, there are specific approaches to obtain the text descriptions. In zero-shot video event detection [21], the text descriptions of the events can be obtained from the event kits provided in the dataset. After obtaining the text descriptions for each class, the next step is to construct the semantic space and generate class prototypes from these descriptions. The semantic space is usually constructed by the keywords extracted from these text descriptions, with each dimension corresponding to a keyword. To construct prototypes of the classes, different approaches are proposed. Some works use the binary occurrence indicator [120] or Bag of Words (BOW) representation [4] for each text description. On the other hand, some works exploit information retrieval techniques. For example, [8, 32, 178] use term frequency inversed document frequency (TF-IDF) to represent each class, whereas [31] applies Clustered Latent Semantic Indexing algorithm on the TF-IDF vectors to obtain representation vectors with reduced dimensions.

Some problem-specific spaces. Some engineered semantic spaces are designed specifically for certain problems. For example, in zero-shot character recognition [81], the classes are restricted to be alphanumeric characters. The semantic space in [81] consists of "canonical" representations of the characters, i.e., a 7×5-pixel image of each character. In problems related to image classification, human gaze data when looking at the images are used [72]. Features extracted from gaze data are used to form the semantic space. In some zero-shot learning problems in computational biology, such as identifying whether a molecular component is active in the presence of a new biological agent [81], descriptions of the biological agents are used to form the semantic space.

Summary of engineered semantic spaces. The advantage of engineered semantic spaces is the flexibility to encode human domain knowledge through the construction of semantic space and class prototypes. The disadvantage is the heavy reliance on humans to perform the semantic space and class prototype engineering. For example, in attribute space, attribute design is needed to be done manually, which needs the domain experts to pay a lot of effort.

2.2 Learned Semantic Spaces

In learned semantic spaces, the dimensions in the spaces are not designed by humans. Prototypes of each class are obtained from the output of some machine learning models. In these prototypes, each dimension does not have an explicit semantic meaning. Instead, the semantic information is contained in the whole prototype. The models used to extract the prototypes can be pretrained in other problems, or trained specifically for the zero-shot learning problem. Next, we introduce typical learned semantic spaces used in zero-shot learning.

Label-embedding spaces. Label-embedding spaces are kinds of semantic spaces in which the class prototypes are obtained through the embedding of class labels. This kind of space is introduced in view of the development and wide utilization of word embedding techniques in natural language processing. In word embedding, words or phrases are embedded into a real number space as vectors. This embedding space contains semantic information. In this space, semantically similar words or phrases are embedded as nearby vectors, while semantically dissimilar words or phrases are embedded as vectors far apart. In zero-shot learning, for each class, the class label of it is a word or a phrase. Thus, it can be embedded into a word embedding space, with the obtained corresponding vector as the prototype. In existing works, different embedding techniques, such as Word2Vec [4, 144, 154] and GloVe [4, 154], have been adopted. Besides, different corpora, from general ones such as Wikipedia [135, 144], to specific ones such as texts from Flickr [14], have been used to learn the embedding model. In addition to generating one prototype for each class, there are also works [103, 125] that generate more than one prototype for each class in the label embedding space. In these works, the prototypes of a class are usually multiple vectors following Gaussian distribution.

Text-embedding spaces. Text-embedding spaces are a kind of semantic spaces, in which the class prototypes are obtained by embedding the text descriptions for each class. Being similar to text-keyword spaces, the semantic information in text-embedding spaces also comes from text descriptions. However, there exists a major difference between these two kinds of spaces. Specifically, a text-keyword space is constructed through extracting keywords and using each of them as a dimension in the constructed space. A text-embedding space is constructed through some learning models. The text descriptions of each class are used as the input of the learning model, and the output vectors are regarded as the prototypes of this class. For example, in an image object recognition task [124], several text descriptions are collected for each class. These text descriptions are used as input of a text encoder model, and the output vectors are regarded as the class prototypes.

Image-representation spaces. Image-representation spaces are kinds of semantic spaces in which the class prototypes are obtained from images belonging to each class. For example, in a video action recognition task [146], images of different action classes are obtained through the search engines. Then, for each action class, images belonging to this class are used as input to some pretrained model (e.g., GoogLeNet pretrained on the ImageNet dataset). The output vectors from the model are combined to form a representation vector and is used as the prototype of this action class.

Summary of learned semantic spaces. The advantage of learned semantic spaces is that the process of generating them is relatively less labour intensive, and the generated semantic spaces contain information that can be easily overlooked by humans. The disadvantage is that, the prototypes of classes are obtained from some machine learning models, and the semantics of each dimension are implicit. In this way, it is inconvenient for humans to incorporate domain knowledge about the classes into the prototypes.

3 METHODS

We classify existing zero-shot learning methods into two categories: *classifier-based methods* and *instance-based methods*. For classifier-based methods, the focus is on how to directly learn a classifier for the unseen classes. For instance-based methods, the focus is on how to obtain labeled instances belonging to the unseen classes and use them for classifier learning. We summarize a method categorization hierarchy in Figure 2b. Next, we introduce each category in this method hierarchy. When introducing the methods, we mainly focus on the most standard data setting of zero-shot learning. That is, for a zero-shot learning task, we consider one semantic space and represent each class with one prototype in that space. Specifically, we will discuss existing works using multiple semantic spaces or multiple prototypes for each class in the semantic space in Section 3.3.

3.1 Classifier-Based Methods

According to the approaches to construct classifiers, we further classify the classifier-based methods into three sub-categories: (1) *correspondence methods*, (2) *relationship methods*, and (3) *combination methods*. Existing classifier-based methods usually take a one-vs-rest solution for learning the multi-class zero-shot classifier $f^u(\cdot)$. That is, for each unseen class c_i^u, they learn a binary one-vs-rest classifier. We denote $f_i^u(\cdot) : \mathbb{R}^D \rightarrow \{0, 1\}$ as the binary one-vs-rest classifier for class $c_i^u \in \mathcal{U}$. Therefore, the eventual zero-shot classifier $f^u(\cdot)$ for the unseen classes consists of N_u binary one-vs-rest classifiers $\{f_i^u(\cdot)|i = 1, \ldots, N_u\}$. Next, we introduce each subcategory with insight and methods under different learning settings.

3.1.1 Correspondence Methods. Its **insight** is to construct the classifier for unseen classes via the correspondence between the binary one-vs-rest classifier for each class and its corresponding class prototype.

In the semantic space, for each class, there is just one corresponding prototype. Thus, this prototype can be regarded as the "representation" of this class. Meanwhile, in the feature space, for each class, there is a corresponding binary one-vs-rest classifier, which can also be regarded as the "representation" of this class. Correspondence methods aim to learn a *correspondence function* between these two types of "representations". In correspondence methods, the correspondence function $\varphi(\cdot; \varsigma)$ takes the prototype $\mathbf{t}_i$ of a class c_i as input, and outputs the parameter ω_i of the binary one-vs-rest classifier $f_i(\cdot; \omega_i)$ for this class, i.e., $\omega_i = \varphi(\mathbf{t}_i; \varsigma)$. After obtaining this correspondence function, for an unseen class c_i^u, with its prototype $\mathbf{t}_i^u$, the corresponding binary one-vs-rest classifier $f_i^u(\cdot)$ can be constructed.

The **general procedure** of correspondence methods is as follows. First, with the available data (under different learning settings, the available data are different), the correspondence function $\varphi(\cdot; \varsigma)$ is learned. Then, for each unseen class c_i^u, with its prototype $\mathbf{t}_i^u$ and the learned correspondence function $\varphi(\cdot; \varsigma)$, the binary one-vs-rest classifier $f_i^u(\cdot)$ is constructed. Finally, with these binary classifiers $\{f_i^u(\cdot)\}_{i=1}^{N_u}$ for unseen classes, classification of the testing instances X^{te} is achieved. In existing works, this procedure can be under different learning settings. Next, we introduce methods under each setting one by one.

Class-Inductive Instance-Inductive (CIII) Setting

• *Typical approach*: In the training phase, with the training instances D^{tr}, a set of binary one-vs-rest classifiers $f^s(\cdot;\omega^s) = \{f_i^s(\cdot;\omega_i^s)\}_{i=1}^{N_s}$ for the seen classes are learned. That is,

$$\min_{\omega^s} \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \ell_1(f^s(\mathbf{x}_i^{tr};\omega^s), y_i^{tr}) + \lambda_1 R_1(\omega^s), \tag{1}$$

where ℓ_1 is a loss function, $R_1(\omega^s)$ is the regularization term (such as the $L1$ or $L2$ norm) of the parameters ω^s, and λ_1 is a tradeoff parameter. Then, for each seen class c_i^s, a pair of binary classifier parameter and class prototype $(\omega_i^s, \mathbf{t}_i^s)$ can be obtained. With the pairs $\{(\omega_i^s, \mathbf{t}_i^s)\}_{j=1}^{N_s}$ from all seen classes, the correspondence function $\varphi(\cdot;\varsigma)$ is learned by

$$\min_{\varsigma} \frac{1}{N_s} \sum_{i=1}^{N_s} \ell_2(\varphi(\mathbf{t}_i^s;\varsigma), \omega_i^s) + \lambda_2 R_2(\varsigma), \tag{2}$$

where ℓ_2 is a loss function, $R_2(\varsigma)$ is the regularization term of the parameter ς, and λ_2 is a tradeoff parameter. In the testing phase, with the learned correspondence function $\varphi(\cdot)$ and the unseen class prototypes T^u, binary one-vs-rest classifiers $\{f_i^u(\cdot)\}_{i=1}^{N_u}$ for the unseen classes are constructed. Classification of the testing instances X^{te} is achieved with these binary classifiers.

• *Different implementations*: Existing works implement Equations (1) and (2) in different ways. An example of strictly implementing the above typical approach is GFZSL (in CIII setting) [142]. It implements Equation (1) with exponential family classifiers and implements Equation (2) with regression functions.

Most existing works go beyond the above typical approach. The rationale is that, in the typical approach, the binary classifiers in Equation (1) and the correspondence function in Equation (2) are learned independently. If they can be learned together, the learning processes may benefit from each other. Therefore, unified optimization frameworks which aim to learn the *compatibility* between pairs of instance and prototype have been proposed, with the objective function in the form of

$$\min_{\varsigma} \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \ell_{c_j^s \in \mathcal{S}}(\Delta(y_i^{tr}, c_j^s), \theta(\mathbf{x}_i^{tr}, \pi(c_j^s);\varsigma), \theta(\mathbf{x}_i^{tr}, \pi(y_i^{tr});\varsigma)) + \lambda R(\varsigma), \tag{3}$$

or

$$\min_{\varsigma} \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \ell_{c_j^s \in \mathcal{S}}(\phi(y_i^{tr}, c_j^s), \theta(\mathbf{x}_i^{tr}, \pi(c_j^s);\varsigma)) + \lambda R(\varsigma), \tag{4}$$

where $\Delta(\cdot,\cdot)$ and $\phi(\cdot,\cdot)$ are usually functions denoting the relation between y_i^{tr} and c_j^s, ℓ is the loss function, $R(\varsigma)$ is the regularization term of the parameter ς, λ is the tradeoff parameter. Finally, $\theta(\cdot,\cdot;\varsigma) : \mathbb{R}^D \times \mathbb{R}^M \rightarrow \mathbb{R}$ is a *compatibility function* [2, 155] that takes an instance $\mathbf{x}_i$ and a class prototype $\mathbf{t}_j$ as inputs, and outputs a compatibility score denoting the confidence/probability about $\mathbf{x}_i$ belonging to class c_j. In Equation (3), the loss function ℓ is optimized for the purpose that the compatibility score of an instance with the prototype of its corresponding class is higher than with the prototypes of other seen classes. In Equation (4), the loss function ℓ is optimized without the comparison as in Equation (3). It is optimized for the purpose that the compatibility score for an instance with the prototype of its corresponding class is high, and that with the prototypes of other seen classes are low.

Given the learned compatibility function θ, for an unseen class c_j^u, with the corresponding prototype $\mathbf{t}_j^u$, the corresponding binary classifier is constructed as $f_j^u(\cdot) = \theta(\cdot, \mathbf{t}_j^u;\varsigma)$. Thus, in this view, the compatibility function acts as the correspondence function. There are various ways to define $\theta(\cdot,\cdot;\varsigma)$, and one popular choice is the bilinear function $\theta(\mathbf{x}_i, \mathbf{t}_j; W) = \mathbf{x}_i^T W \mathbf{t}_j$, where

$\varsigma = W \in \mathbb{R}^{D \times M}$ is the parameter. Such bilinear function $\theta(\mathbf{x}_i, \mathbf{t}_j; W)$ is widely used in many methods, including DeViSE [38], ALE [2], SJE [4], ESZSL [131] and DSSC [90]. It is worth noting that, the binary one-vs-rest classifiers obtained through this function are effectively linear classifiers (for class c_j, the parameter of the binary one-vs-rest classier is $W\mathbf{t}_j$). As the classification ability of linear classifiers is limited, there are some works exploiting some sophisticated definitions for $\theta(\cdot, \cdot)$ as an extension of the bilinear function. For example, $\theta(\cdot, \cdot)$ is defined as a piecewise bilinear function in LatEm [154], a combination of a nonlinear projection function and a bilinear function in NCBM [148], and the parameter W as a multiplication of two matrices in [32, 120].

There are also other choices of the compatibility function: (1) $\theta(\mathbf{x}_i, \mathbf{t}_j) = \phi(\mathbf{x}_i)^T \xi(\mathbf{t}_j)$, where $\phi(\cdot)$ and $\xi(\cdot)$ are projection functions, such as linear projection functions in [165], or nonlinear projection functions in [8, 168]; (2) $\theta(\mathbf{x}_i, \mathbf{t}_j) = \phi(\mathbf{x}_i)^T W \xi(\mathbf{t}_j)$, where $\phi(\cdot)$ and $\xi(\cdot)$ are also projection functions [174]; (3) instance $\mathbf{x}_i$ and class prototype $\mathbf{t}_j$ are concatenated into a vector and put into a binary classifier to decide whether they come from the same class (e.g., "input space view" model in [81] and method in [117]). This binary classifier acts as the compatibility function; and (4) the compatibility function in other forms (e.g., the compatibility function in methods LinDisc-1-vs-all, LinDisc-0-1, and NNet-0-1 [81]).

Class-Transductive Instance-Inductive (CTII) Setting

• *Typical approach*: In the training phase, with D^{tr}, T^s and T^u, the correspondence function $\varphi(\cdot)$ and the classifier $f^u(\cdot)$ for unseen classes are learned together as

$$\varphi(\cdot), f^u(\cdot) = \phi(D^{tr}, T^s, T^u), \tag{5}$$

where ϕ is the learning process. In the testing phase, with the learned classifier $f^u(\cdot)$, classification of the testing instances X^{te} is achieved.

• *Different implementations*: The learning process ϕ can be implemented in different ways. In some methods, with D^{tr} and T^s, some initial correspondence functions are firstly learned. Then with D^{tr} and T^u, the final correspondence function is learned [31]. In some other methods, the correspondence function is learned with D^{tr}, T^s and T^u in a unified process (e.g., method in [150] and BZ-SCR [119]).

Class-Transductive Instance-Transductive (CTIT) Setting

• *Typical approach*: With D^{tr}, T^s, X^{te} and T^u, through the learning process

$$\varphi(\cdot), f^u(\cdot), Y^{te} = \phi(D^{tr}, T^s, X^{te}, T^u), \tag{6}$$

the correspondence function $\varphi(\cdot)$, the classifier $f^u(\cdot)$ for unseen classes, and the labels Y^{te} of testing instances X^{te} can be obtained. ϕ in Equation (6) is the learning process.

• *Different implementations*: The learning process ϕ can be implemented in different ways. In some methods, the correspondence function is first learned with D^{tr} and T^s, and then adjusted with X^{te} and T^u (e.g., GFZSL (in CTIT setting) [142] and DMaP [91]). In some methods, the correspondence function is learned with D^{tr}, T^s, X^{te} and T^u in a unified process (e.g., SMS [60]). In some other methods, to improve the classification performance on testing instances X^{te}, class prototypes (T^s and T^u) are also adjusted during model optimization [88].

3.1.2 Relationship Methods. Its **insight** is to construct classifier for the unseen classes based on the relationships among classes.

In the feature space, binary one-vs-rest classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ for the seen classes can be learned with the available data (under different learning settings, the available data are different). Meanwhile, the relationships among the seen and the unseen classes can be obtained by calculating the relationships among corresponding prototypes or by other approaches. Relationship methods aim

to construct the classifier $f^u(\cdot)$ for the unseen classes through these learned binary seen class classifiers and these class-relationships.

The **general procedure** of relationship methods is as follows. First, binary one-vs-rest classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ for the seen classes $\mathcal{S}$ are learned with the available data. Then, the relationships δ among the seen and the unseen classes are calculated via the corresponding class prototypes or obtained through other approaches. Finally, with these classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ and the relationships δ, classifier $f^u(\cdot) = \{f_i^u(\cdot)\}_{i=1}^{N_u}$ for the unseen classes $\mathcal{U}$ is constructed and classification of the testing instances X^{te} is achieved. In existing works, this procedure can be under different learning settings. Next, we introduce methods under different settings one by one.

Class-Inductive Instance-Inductive (CIII) Setting

• *Typical approach*: In the training phase, with the training instances D^{tr}, classifier $f^s(\cdot) = \{f_i^s(\cdot)\}_{i=1}^{N_s}$ for the seen classes $\mathcal{S}$ can be learned. That is,

$$\min_{\varsigma} \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \ell(f^s(\mathbf{x}_i^{tr}; \varsigma), y_i^{tr}) + \lambda R(\varsigma), \tag{7}$$

where ℓ is a loss function, $R(\varsigma)$ is the regularization term of the parameter ς, and λ is a tradeoff parameter. After obtaining the classifier $f^s(\cdot)$ for the seen classes, in the testing phase, for each unseen class c_i^u, the corresponding binary classifier $f_i^u(\cdot)$ can be constructed as the weighted sum of binary classifiers for some selected seen classes. That is,

$$f_i^u(\cdot) = \sum_{j=1}^{K} \delta(c_i^u, c_j^s) \cdot f_j^s(\cdot), \tag{8}$$

where K is the number of seen classes selected to construct the classifier $f_i^u(\cdot)$, $\delta(c_i^u, c_j^s)$ is the relationship between the seen class c_j^s and the unseen class c_i^u. From the perspective of probability, Equation (8) can be interpreted as

$$p(c_i^u|\mathbf{x}^u) = \sum_{j=1}^{K} p(c_i^u|c_j^s) \cdot p(c_j^s|\mathbf{x}^u), \tag{9}$$

where given a testing instance $\mathbf{x}^u$, its probabilities of belonging to the seen classes are first calculated, and then the transition probabilities from the seen classes to the unseen class c_i^u are calculated. The probability $p(c_j^s|\mathbf{x}^u)$ of belonging to seen class c_j^s is obtained through the binary classifier $f_j^s(\cdot)$. The transition probability from the seen class c_j^s to the unseen class c_i^u is obtained through the relationship $\delta(c_i^u, c_j^s)$ between them.

• *Different implementations*: There are different ways to implement the above approach. In many methods, the process of learning binary classifiers for seen classes in Equation (7) is the same as the process of learning classifiers in supervised learning. As a result, the classifiers learned are usually common classifiers widely used in supervised learning, such as SVM [129] and neural networks [65, 77]. In the process of deriving unseen class classifiers from seen class classifiers, the relationship between each pair of seen and unseen classes is usually obtained from the corresponding prototypes in the semantic space [48, 65, 77], such as through calculating the cosine similarities between them [48, 77]. In addition, there are other ways to obtain the class relationships, such as using the ontological structures in WordNet [48, 129], or the hit-count of web search results [129]. For each unseen class, the K relevant seen classes can also be selected based on different strategies [77], such as the top K predicted seen classes by seen class classifiers, or the K most similar classes under the measurement of relationships.

Instead of just using the training instances D^{tr} to learn binary seen class classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$, there are also methods in which the process of learning seen class classifiers involves seen class prototypes T^s. For example, in SSE [173], in the training phase, each seen class is represented as a combination of all seen classes. With the training instances, binary classifiers for seen classes are learned. In the testing phase, for an unseen class, with the prototypes in the semantic space, the relationships of it with all seen classes are calculated. Then, classifier for this unseen class is obtained.

To increase the flexibility when obtaining unseen class classifier $f^u(\cdot)$ via class relationships δ, in some methods [16], some phantom classes $C^{\mathcal{PH}} = \{c_i^{ph}\}_{i=1}^{N_{ph}}$ are introduced. These phantom classes do not have real instances in the feature space. However, similar to the seen and the unseen classes, each phantom class c_i^{ph} has a prototype $\mathbf{t}_i^{ph}$ in the semantic space and a binary one-vs-rest classifier $f_i^{ph}(\cdot)$ in the feature space. In the training phase, with D^{tr} and T^s, the binary classifier $f_i^{ph}(\cdot)$ and prototype $\mathbf{t}_i^{ph}$ for each phantom class c_i^{ph} is obtained. In the testing phase, with the prototypes of the phantom and the unseen classes, and the binary classifiers for the phantom classes, the binary classifiers $\{f(\cdot)_i^u\}_{i=1}^{N_u}$ for the unseen classes $\mathcal{U}$ are constructed.

Class-Transductive Instance-Inductive (CTII) Setting

• *Typical approach*: In the training phase, with the training instances D^{tr}, binary classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ for the seen classes are first learned. Then, with the prototypes T^s of the seen classes and the prototypes T^u of the unseen classes, a graph $\mathcal{G}$ is constructed by taking these classes as nodes. In this way, relationships δ among classes can be obtained via this graph. With the relationships δ in $\mathcal{G}$ and the learned binary seen class classifiers $\{f_j^s(\cdot)\}_{j=1}^{N_s}$, binary classifiers $\{f_i^u(\cdot)\}_{i=1}^{N_u}$ for the unseen classes $\{c_i^u\}_{i=1}^{N_u}$ can be obtained by

$$\{f_i^u(\cdot)\}_{i=1}^{N_u} = g(f_1^s(\cdot), f_2^s(\cdot), \ldots, f_{N_s}^s(\cdot), \delta), \tag{10}$$

where g is the function generating these classifiers. In the testing phase, with the obtained binary unseen class classifiers, classification of the testing instances X^{te} is achieved.

• *Different implementations*: Different methods adopt different approaches to construct the graph $\mathcal{G}$, and accordingly, they adopt different ways to use this graph to construct the binary classifiers $\{f_i^u(\cdot)\}_{i=1}^{N_u}$ for the unseen classes. For example, in [45, 46], a directed kNN graph $\mathcal{G}$ with the seen and the unseen classes as nodes is constructed in the semantic space. The weight of each edge connecting two classes in the graph is based on the distance between their corresponding prototypes. For a testing instance $\mathbf{x}_i^u$, by using the learned binary seen class classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$, the probabilities of belonging to each seen class are obtained. Based on these probabilities, this instance is connected to K seen classes with the highest probabilities. After that, absorbing Markov chain process on the graph $\mathcal{G}$ is performed to obtain the label for this instance. In this method, for an unseen class c_i^u, the binary seen class classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ together with the inference from seen classes to this class on the graph $\mathcal{G}$ form the binary classifier $f_i^u(\cdot)$ for this class. Another example is the method in [25], which is suitable for binary attribute spaces. It uses relationships among classes in a hierarchy and exclusion (HEX) graph. In this method, a HEX graph with attributes and class labels (both the seen and the unseen classes) as nodes is constructed. Similarly, for an unseen class c_i^u, the binary seen class classifiers $\{f_i^s(\cdot)\}_{i=1}^{N_s}$ together with the inference from the seen classes to this class on the graph $\mathcal{G}$ form the classifier $f_i^u(\cdot)$ for this class.

Class-Transductive Instance-Transductive (CTIT) Setting

• *Typical approach*: With D^{tr}, T^s, X^{te} and T^u, through the learning process

$$f^s(\cdot), f^u(\cdot), Y^{te} = \phi(D^{tr}, T^s, X^{te}, T^u), \tag{11}$$

the classifier $f^s(\cdot)$ for seen classes, the classifier $f^u(\cdot)$ for unseen classes and the labels Y^{te} of testing instances X^{te} can be obtained. ϕ in Equation (11) is the learning process.

• *Different implementations*: The learning process ϕ can be implemented in different ways. In some methods [176], the unseen class classifier $f^u(\cdot)$ is firstly learned with D^{tr}, T^s and T^u. Then, this classifier and the labels Y^{te} of testing instances X^{te} are adjusted with the unseen class prototypes T^u and the testing instances X^{te}. In some methods [163], when learning classifier $f^s(\cdot)$ for the seen classes, the testing instances X^{te} are also involved. In some methods [175], the classification of testing instances X^{te} is not only by the learned unseen class classifier $f^u(\cdot)$, but also by the structural information of the testing instances. (This classification approach can also be combined with unseen class classifiers obtained through methods in other categories, such as methods in [131] and [174].)

3.1.3 Combination Methods. Its **insight** is to construct the classifier for unseen classes by the combination of classifiers for basic elements which are used to constitute the classes.

In combination methods, it is regarded that there is a list of "basic elements" to constitute the classes. Each of the seen and the unseen classes is a combination of these basic elements. Embodied in the semantic space, it is regarded that each dimension represents a basic element, and each class prototype denoting the combination of these basic elements for the corresponding class. Thus, methods in this category are mainly suitable for semantic spaces in which each dimension of the class prototypes takes the value of either 1 or 0, denoting whether a class has the corresponding element or does not. Typical examples of these kinds of spaces are binary attribute spaces, some lexical spaces and some text-keyword spaces. As existing methods in this category are mainly developed based on the binary attribute spaces, in the following, when we refer to "basic elements", we use the term "attributes" instead. In binary attribute spaces, the prototypes of seen and unseen classes consist of attributes, with each dimension being an attribute. Thus, in this section, when referring to the prototype $\mathbf{t}_i$ of a class c_i, we use $\mathbf{a}_i$ instead to denote that the prototype consists of attributes. Accordingly, the prototypes of the seen classes are denoted as $A^s = \{\mathbf{a}_i^s\}_{i=1}^{N_s}$, and the prototypes of the unseen classes are denoted as $A^u = \{\mathbf{a}_i^u\}_{i=1}^{N_u}$.

The **general procedure** of combination methods is as follows. With the available data (the available data are different under different learning settings), classifiers $\{f_i^a(\cdot)\}_{i=1}^M$ for the attributes are firstly learned. Then, with the learned classifiers for the attributes, classifier $f^u(\cdot) = \{f_i^u(\cdot)\}_{i=1}^{N_u}$ for the unseen classes is obtained through some inference model. In this paper, we refer to the inference model from attribute classifiers to unseen class classifier as the *inference framework*, denoted as $Fmk()$. The above procedure can be under different learning settings. Most of existing methods are under the CII setting, and a few of them are under the CTIT setting. To the best of our knowledge, we do not see existing methods under the CTII setting. Next, we introduce methods under different settings one by one.

Class-Inductive Instance-Inductive (CIII) Setting

• *Typical approach*: In the training phase, with the training instances D^{tr} and the seen class prototypes A^s, classifiers $\{f_i^a(\cdot)\}_{i=1}^M$ for the attributes are learned. Then, in the testing phase, with the attribute classifiers and some inference framework $Fmk()$, for unseen classes $\{c_i^u\}_{i=1}^{N_u}$, with the corresponding prototypes $\{\mathbf{a}_i^u\}_{i=1}^{N_u}$, the classifier $f^u(\cdot) = \{f_i^u(\cdot)\}_{i=1}^{N_u}$ is

$$f^u(\cdot) = Fmk(f_1^a(\cdot), f_2^a(\cdot), \ldots, f_M^a(\cdot), \{\mathbf{a}_i^u\}_{i=1}^{N_u}). \tag{12}$$

• *Different implementations*: In the method DAP in the pioneer work [80], the attribute classifiers $\{f_i^a(\cdot)\}_{i=1}^M$ are SVMs. To learn the classifier $f_i^a(\cdot)$ for each attribute $a_{(i)}$, training instances from classes that have this attribute constitute the positive instances, training instances from classes

that do not have this attribute constitute the negative instances. With these instances, the classifier $f_i^a(\cdot) : \mathcal{X} \rightarrow \{0, 1\}$ for attribute $a_{(i)}$ is learned. After learning the attribute classifiers, the inference from attribute to unseen class is represented by the inference framework in a probability form

$$p(c_i^u|\mathbf{x}_j^{te}) = \sum_{\mathbf{a}\in\{0,1\}^M} p(c_i^u|\mathbf{a})p(\mathbf{a}|\mathbf{x}_j^{te}) = \frac{p(c_i^u)}{p(\mathbf{a}_i^u)} \prod_{m=1}^{M} p(a_{i,(m)}^u|\mathbf{x}_j^{te}), \tag{13}$$

where $p(c_i^u|\mathbf{x}_j^{te})$ is the probability that the testing instance $\mathbf{x}_j^{te}$ belongs to the unseen class c_i^u, $p(c_i^u)$ is the prior probability of unseen class c_i^u, $a_{i,(m)}^u$ is the value of the mth attribute for the prototype of the unseen class c_i^u, $p(a_{i,(m)}^u|\mathbf{x}_j^{te})$ is the probability of instance $\mathbf{x}_j^{te}$ having the attribute value $a_{i,(m)}^u$ that class c_i^u has (1 or 0) for the mth attribute, and $p(\mathbf{a}_i^u) = \prod_{m=1}^{M} p(a_{i,(m)}^u)$ is the probability of the prototype $\mathbf{a}_i^u$. In this inference framework, given a testing instance, the probabilities of each unseen class are obtained from the above multiplication. We refer to framework in this form as the *multiplication framework*. In [130], a similar multiplication framework is proposed.

From this pioneer work, methods in this form have been proposed. Some of them focus on improving the classification ability of attribute classifiers. Some others focus on modifying the inference framework. Next, we introduce methods from these two aspects.

Methods focusing on improving attribute classifiers. To improve the classification ability of attribute classifiers, some methods that aim to improve the generalization ability of attribute classifiers are proposed. For example, KDICA [50] aims to improve the generalization ability of attribute classifiers across different classes by regarding each class as a domain, and proposing a mechanism to learn new representations of instances. The method in [67] treats the problem of attribute classification as a multi-task learning problem (with each attribute as a task), and aims to select the relevant features for each attribute.

Some other methods aim to improve the classification ability of attribute classifiers by involving additional information, such as other attributes or seen class labels. (1) *Methods involving information about other attributes.* In HAP [62], a hypergraph which contains information about attribute relations in the training instances is constructed, and the attribute prediction problem is converted to a regularized hypergraph cutting problem. (2) *Methods involving information about class labels.* In [62], by further incorporating class label information into the hypergraph, it forms another method CSHAP, which considers not only information about other attributes, but also information about class labels. In IAP [80], classifiers for seen classes are learned. The classifier for each attribute is the combination of classifiers for seen classes which contain this attribute. UMF [96] incorporates class label information into the process of attribute classifier learning. BAP [86] is a combination of DAP and IAP, with different combination strategies to combine classifiers. (3) *Methods involving other information.* HAT [5] utilizes the hierarchical structure in WordNet to learn attribute classifiers for each class separately (in this method, the inference framework is not a multiplication framework, but one that calculates the arithmetic average of probabilities for attributes).

Methods with other inference frameworks. Besides improving attribute classifiers, there are also methods focusing on improving the inference framework. The method in [139] considers the importance (called dominance in this work) of different attributes when inferring class labels from attribute classification results. The method in [149] uses a Bayesian network to capture the relationships among attributes and classes. In this way, the relationships among attributes are utilized in attribute-to-class inference.

Class-Transductive Instance-Transductive (CTIT) Setting

- *Typical approach*: With D^{tr}, A^s, X^{te} and A^u, through the learning process

$$\{f_i^a(\cdot)\}_{i=1}^M, f^u(\cdot), Y^{te} = \phi(D^{tr}, A^s, X^{te}, A^u), \tag{14}$$

the classifiers $\{f_i^a(\cdot)\}_{i=1}^M$ for attributes, the classifier $f^u(\cdot)$ for unseen classes and the labels Y^{te} of testing instances X^{te} can be obtained. ϕ is the learning process.

• *Different implementations*: The learning process ϕ can be implemented in different ways, in which the transduction is embodied in different phases. Some methods embody the transduction in the process of attribute classifier learning. For example, the method in [52] uses testing instances when learning attribute classifiers, by forming a graph containing both the training instances and the testing instances. Some methods embody the transduction in the process of testing instance classification. For example, in PST [128], label propagation (which is a widely used approach in semi-supervised learning) is performed on the kNN graph formed by the testing instances. This classification process utilizes the relationships among testing instances when predicting their labels.

3.2 Instance-Based Methods

Instance-based methods aim to firstly obtain labeled instances for the unseen classes, then with these instances to learn the zero-shot classifier $f^u(\cdot)$. According to the source of these instances, existing instance-based methods can be classified into three subcategories: (1) *projection methods*, (2) *instance-borrowing methods*, and (3) *synthesizing methods*. Next, we introduce each subcategory with insight and methods under different learning settings.

3.2.1 Projection Methods. Its **insight** is to obtain labeled instances for the unseen classes by projecting both the feature space instances and the semantic space prototypes into a common space.

In the feature space, there are labeled training instances belonging to the seen classes. Meanwhile, in the semantic space, there are prototypes of both the seen and the unseen classes. Both of the feature space and the semantic space are real number spaces, with instances and prototypes as vectors in them. In this view, the prototypes can also be regarded as labeled instances. Thus, we have labeled instances in two spaces (the feature space $\mathcal{X}$ and the semantic space $\mathcal{T}$). In projection methods, instances in these two spaces are projected into a common space. In this way, we can obtain labeled instances belonging to the unseen classes. In this paper, we refer to this common space as the *projection space*, denoted as $\mathcal{P}$.

The **general procedure** of projection methods is as follows. Firstly, instances $\mathbf{x}_i$ in the feature space $\mathcal{X}$ and prototypes $\mathbf{t}_j$ in the semantic space $\mathcal{T}$ are projected into the projection space $\mathcal{P}$ with the projection functions $\theta(\cdot)$ and $\xi(\cdot)$ respectively:

$$\mathcal{X} \rightarrow \mathcal{P} : \mathbf{z}_i = \theta(\mathbf{x}_i), \tag{15}$$

$$\mathcal{T} \rightarrow \mathcal{P} : \mathbf{b}_j = \xi(\mathbf{t}_j). \tag{16}$$

Then, classification is performed in the projection space.

In projection methods, for each unseen class c_i^u, it does not have labeled instances in the feature space, thus its prototype $\mathbf{t}_i^u$ in the semantic space is the only labeled instance belonging to this class. That is, for each unseen class, there is only one labeled instance available. It is difficult to learn classifiers like SVM or logistic regression classifier with so few labeled instances for the unseen classes. As a result, in existing projection methods, the classification is usually performed by *nearest neighbor classification (1NN)* or some variants of it. The 1NN method can work in situations where just one labeled instance is available for the classes needed to classify, which is just suitable for this scenario. The 1NN method is a kind of lazy learning method, with no explicit learning process needed. Thus, when use it for classification, no explicit process of learning classifiers for the unseen classes is needed. The projection methods in existing works can be under different learning settings. Next, we introduce them one by one.

Class-Inductive Instance-Inductive (CIII) Setting

• *Typical approach*: In the training phase, with the training instances D^{tr} and the seen class prototypes T^s, the projection functions θ and ξ are learned. In the testing phase, with the learned projection functions, testing instances X^{te} and unseen class prototypes T^u are projected into the projection space $\mathcal{P}$. Then, each projected testing instance is classified to the nearest unseen class prototype via 1NN classification.

• *Different implementations*: The projection spaces in existing works can be diverse. They can be the semantic space, the feature space, another common space and more than one space. We introduce methods adopting different projection spaces one by one.

Semantic space as projection space. In methods that adopt the semantic space as projection space, instances in the feature space are projected into the semantic space. Thus, projection function ξ in Equation (16) is an identity function, and classification is performed in the semantic space. For methods in this category, the classification method is similar (in most of them are 1NN or some variants of it), but the projection functions are diverse.

In some methods [102, 113, 135], a *regression function* is used as the projection function θ. To learn the regression function, the objective function is often in the form of

$$\min_{\varsigma} \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \ell(\theta(\mathbf{x}_i^{tr}; \varsigma), \pi(y_i^{tr})) + \lambda R(\varsigma), \tag{17}$$

where the projection function $\theta(\cdot; \varsigma)$ is a regression function with parameter ς. ℓ is a loss function; usually it is a square loss function. $R(\varsigma)$ is the regularization term of parameter ς, and λ is a tradeoff parameter. The regression function can be a linear regression function [82, 102, 113] or a nonlinear regression function [82, 135].

Despite the success of Equation (17) in many zero-shot learning problems, it suffers from a problem of *hubness*. As introduced in [29], in projection methods, when adopting the semantic space as projection space and regression as the projection function, after projecting feature space instances into the semantic space $\mathcal{T}$, some prototypes are nearest neighbors of a large number of instances from different classes. These prototypes are referred to as "hubs", and this phenomenon is referred to as "hubness".

The existence of hubs often degrades the performance of classification. There have been methods proposed from different aspects to relieve its effect. Some methods use other projection spaces [134, 171], while some others are under the CTIT learning setting [29]. We will introduce these methods in the following. In the setting of CIII and adopting the semantic space as projection space, some methods are proposed with different loss functions. For example, in [83] the loss function is a margin-based ranking loss and distances between instances are measures by cosine similarity. This work adopts a different loss function mainly to overcome the problem caused by square loss. In this way, the variance of the projected instances is increased, and the effect of hubness is relieved.

In [20, 33, 62, 97, 170], the projection function θ consists of *classifiers for attributes*. Being similar to combination methods introduced in Section 3.1.3, this kind of projection function is mainly suitable for binary attribute spaces (an exception is [170], which uses a continuous semantic space). The process of attribute prediction is the same as the process of attribute prediction in combination methods. However, in this kind of methods, the predicted probabilities by attribute classifiers are treated as coordinate values of the projected instances in the semantic space, and 1NN classification is performed.

In some methods, the projection θ is achieved by *convex combination of seen class prototypes*. In ConSE [110], an n-way classifier $f^s(\cdot)$ for seen classes is firstly learned with training instances D^{tr}. For a testing instance $\mathbf{x}_i^{te}$, the probabilities of it belonging to each seen class is calculated with this learned classifier $f^s(\cdot)$. With the corresponding calculated probability values as coefficients, the

combination of the prototypes of the top K predicted seen classes is the projection of this instance $\mathbf{x}_i^{te}$ in the semantic space

$$\theta(\mathbf{x}_i^{te}) = \frac{1}{Q}\sum_{j=1}^{K} p(c_j^s|\mathbf{x}_i^{te}) \cdot \mathbf{t}_j^s, \tag{18}$$

where $p(c_j^s|\mathbf{x}_i^{te})$ is the probability of $\mathbf{x}_i^{te}$ belonging to the seen class c_j^s, and Q is a normalization factor. This kind of projection function is also adopted in HierSE [89].

The projection θ can also be achieved by *matrix factorization*. Usually the process is similar to *dictionary learning*. For an instance $\mathbf{x}_i$, the projection is usually in the form of $\|\mathbf{x}_i - B\pi(y_i)\|_2$ where B is the dictionary, $\pi(y_i) \rightarrow \mathbf{t}_i$ is the corresponding prototype of $\mathbf{x}_i$. In different methods, there are different ways to learn this projection function. In DSRL [169], the projection (from the feature space to an extension of the semantic space) is achieved by non-negative matrix factorization. In MFMR [164], the projection is in the form of $\|\mathbf{x}_i - BT^s\mathbf{y}_i\|_2$ in which $\pi(y_i)$ is written as $T^s\mathbf{y}_i$. $\mathbf{y}_i$ is the one-hot form of label y_i. In this method, with manifold regularizations, matrix tri-factorization is used to learn the projection function.

Other projection functions such as Canonical Correlation Analysis (CCA) [82] and Singular Value Decomposition (SVD) [82] are also adopted.

Feature space as projection space. For methods in this category, prototypes in the semantic space are projected into the feature space. Thus, the projection function θ in Equation (15) is an identity function. Classification is performed in the feature space. Compared with the number of methods that adopt the semantic space as projection space, the number of methods in this category is small, and most methods adopt a regression function as the projection function. As mentioned when introducing the "hubness" phenomenon, some methods [134, 171] in this category are proposed to relieve the problem of "hubness". They have an objective function in the form of

$$\min_{\varsigma} \frac{1}{N_{tr}}\sum_{i=1}^{N_{tr}} \ell(\mathbf{x}_i^{tr}, \xi(\pi(y_i^{tr}); \varsigma)) + \lambda R(\varsigma), \tag{19}$$

where the projection function $\xi(\cdot;\varsigma)$ is a regression function with parameter ς, ℓ is a loss function, $R(\varsigma)$ is the regularization term of parameter ς, and λ is a tradeoff parameter. The regression function could be a linear regression [134] or a nonlinear regression [171] function. These methods relieve the problem of hubness by adopting the feature space, rather than semantic space, as the projection space.

The projection ξ from the semantic space to the feature space can also be performed through a latent space. Such as in UVDS [98], a kNN graph of training instances D^{tr} is constructed in the latent space taking into account the structure information of both the feature space and the semantic space. In this way, more information about the structure of both spaces can be leveraged when learning the projection function.

Others. There are also methods [24, 27, 121] in which the projection space is neither the semantic space nor the feature space, but *another space*. There are also methods [68, 76, 162] in which *more than one space* is adopted as the projection space. In these methods, more flexibility to the projection functions can be involved.

Class-Transductive Instance-Inductive (CTII) Setting

- *Typical approach*: In the training phase, with D^{tr}, T^s and T^u, projection functions θ and ξ are learned. Then, in the testing phase, with the learned projection functions, unseen class prototypes T^u and testing instances X^{te} are projected into the projection space. In the projection space, classification is performed by either 1NN or other classification methods.

• *Different implementations*: There are different implementations of this typical approach. In some methods [147], the classification is performed by 1NN, and the unseen class prototypes T^u are involved in learning the projection functions. There also exist methods that do not use 1NN as the classification method. For example, in [66], random forest is used as the classification method.

Class-Transductive Instance-Transductive (CTIT) Setting

• *Typical approaches*: With D^{tr}, T^s, X^{te} and T^u, the projection functions θ and ξ are learned. With these learned projection functions, unseen class prototypes T^u and testing instances X^{te} are projected into the projection space and classification is performed in it.

• *Different implementations*: In existing methods, the transduction can be embodied in the projection function learning process or in the classification process. We introduce strategies on these two aspects one by one.

Transductive strategies in projection function learning. In these strategies, when learning the projection functions, testing instances X^{te} and unseen class prototypes T^u are involved.

Manifold regularization [161]. In this strategy, the structure of the training and the testing instances is utilized when learning the projection function. In [161], a space which is neither the feature space nor the semantic space is adopted as the projection space. Manifold regularization [95] is used when projecting instances from the feature space to the projection space. In this strategy, a kNN graph with both the training instances and the testing instances as nodes is constructed in the feature space. This graph encodes the structure information of the feature space. With this graph, manifold regularized regression is performed.

Matrix factorization with testing instances and unseen class prototypes [75, 164]. This strategy is designed for cases where matrix factorization [94] is adopted as the projection function. Similar to methods that use matrix factorization as the projection function under the CIII setting, the projection is in the form of $\|\mathbf{x}_i - B\pi(y_i)\|_2$ or $\|\mathbf{x}_i - BT\mathbf{y}_i\|_2$. However, the dictionary B is learned with X^{te} and T^u involved (e.g., [75]), or the labels Y^{te} of the testing instances X^{te} are obtained via matrix factorization with X^{te} and T^u (e.g., MFMR-joint [164]).

Transductive strategies in classification. After testing instances X^{te} and unseen class prototypes T^u are projected into the projection space $\mathcal{P}$, i.e., $\{\mathbf{x}_i^{te}\}_{i=1}^{N_{te}} \rightarrow \{\mathbf{z}_i^{te}\}_{i=1}^{N_{te}}$ and $\{\mathbf{t}_i^u\}_{i=1}^{N_u} \rightarrow \{\mathbf{b}_i^u\}_{i=1}^{N_u}$, there are some strategies that consider the whole testing instances when performing classification.

Self-training [40, 42, 147, 160, 161]. This strategy aims to adjust the prototypes of unseen classes with the testing instances when performing 1NN classification. For an unseen class c_i^u, the prototype $\mathbf{b}_i^u$ is adjusted as the mean value of the K nearest testing instances [40, 42, 160, 161]:

$$\hat{\mathbf{b}}_i^u = \frac{1}{K} \sum_{\mathbf{z}^{te} \in NN_K(\mathbf{b}_i^u)}^{K} \mathbf{z}^{te}, \tag{20}$$

where $NN_K(\mathbf{b}_i^u)$ is the K nearest testing instances of the prototype $\mathbf{b}_i^u$ [40, 42, 160, 161]. In [147], the prototype is adjusted as the average of $\mathbf{b}_i^u$ and $\hat{\mathbf{b}}_i^u$, i.e., $(\mathbf{b}_i^u + \hat{\mathbf{b}}_i^u)/2$.

Label propagation [75, 169]. This strategy uses the widely adopted label propagation strategy in semi-supervised learning. In this strategy, the structure of the testing instances (in some methods along with the unseen class prototypes) is utilized when performing the classification. In [75], the unseen class prototypes $\{\mathbf{b}_i^u\}_{i=1}^{N_u}$ and the testing instances $\{\mathbf{z}_i^{te}\}_{i=1}^{N_{te}}$ form a graph in the projection space. The prototypes are viewed as labeled instances and label propagation is performed from them on this graph. In [169], the graph is formed just by the testing instances. The scores about the probabilities of each testing instance belonging to each unseen class are calculated. Based on these scores, label propagation is performed on the graph.

Structured prediction [147]. This strategy is a simplified version of the structured prediction in [175]. In this strategy [147], it is assumed that the labels of nearby instances should be similar. With $\{\mathbf{b}_i^u\}_{i=1}^{N_u}$ as the initial clustering centers, K-means clustering is performed on the testing instances $\{\mathbf{z}_i^{te}\}_{i=1}^{N_{te}}$. Then, one-to-one correspondences between clustering centers and unseen class prototypes are calculated via linear programming. Instances in each cluster are classified to the corresponding unseen class.

Strategies for the hubness problem [29]. As mentioned earlier, when adopting the semantic space as projection space and regression as the projection function, hubness is a problem that degrades the classification performance. Along with the methods introduced under the CIII setting, there are also methods under the CTIT setting to deal with this problem. For example, in [29], after testing instances X^{te} are projected into the semantic space via liner regression, two classification methods are proposed. Method NN_{nrm} normalizes the vector of cosine similarities of each unseen class c_i^u to all instances (testing instances and some auxiliary instances) to length 1. In this way, the similarities of each testing instance to each unseen class are re-weighted. In the globally-corrected (GC) method, the classification is not done by the absolute values of cosine similarity, but by the rank scores of the testing instances to each unseen class.

In existing methods under the CTIT setting, usually more than one transductive strategy is adopted. For example, in [161], it adopts both the manifold regularization and the self-training strategies.

3.2.2 Instance-Borrowing Methods. Its **insight** is to obtain labeled instances for the unseen classes by borrowing from the training instances.

Instance-borrowing methods are based on the similarities between classes. For example, in object recognition in images, suppose we want to learn a classifier for the class "truck", but do not have the corresponding labeled instances. However, we have some labeled instances belonging to classes "car" and "bus". As they are similar objects to "truck", when learning classifier for the class "truck", we can use instances belonging to these two classes as positive instances. This kind of methods follow the way human beings recognize the world [58]. We may have never seen instances belonging to some classes, but have seen instances belonging to some similar classes. With the knowledge of these similar classes, we are able to recognize instances belonging to the unseen classes.

The **general procedure** of instance-borrowing methods is as follows. First, for each unseen class c_i^u, some instances from the training instances are borrowed and assigned the label of this class. Then, with the borrowed instances for all unseen classes, the classifier $f^u(\cdot)$ for the unseen classes is learned, and classification of the testing instances X^{te} is achieved.

In instance-borrowing methods, before borrowing the instances, the unseen classes $\mathcal{U}$ should be determined. Only in this way, we know for which classes to borrow the instances. Thus, the optimization of the model is for predetermined unseen classes, and naturally unseen class prototypes T^u are involved in the optimization process. Thus, instance-borrowing methods cannot be under the CIII setting. According to whether the testing instances X^{te} are involved in model optimization, methods in this category can be under the CTII setting or the CTIT setting. Next, we introduce methods under these two settings.

Class-Transductive Instance-Inductive (CTII) Setting

• *Typical approach*: In the training phase, with D^{tr}, T^s and T^u, for each unseen class c_i^u, some training instances are borrowed and assigned with the label of this class

$$\{\{\hat{\mathbf{x}}_i^{U1}, c_1^u\}_{i=1}^{N_{U1}}, \{\hat{\mathbf{x}}_i^{U2}, c_2^u\}_{i=1}^{N_{U2}}, \ldots, \{\hat{\mathbf{x}}_i^{UN_u}, c_{N_u}^u\}_{i=1}^{N_{UN_u}}\} = g(D^{tr}, T^s, T^u), \tag{21}$$

where $\{\hat{\mathbf{x}}_i^{Uj}, c_j^u\}$ is the i-th borrowed instance for the unseen class c_j^u, and N_{Uj} is the number of borrowed instances for the unseen class c_j^u. g is the function that decides the way of borrowing. Then, with these borrowed instances, classifier $f^u(\cdot)$ for the unseen classes $\mathcal{U}$ is learned. In the testing phase, with the learned classifier $f^u(\cdot)$, classification of the testing instances X^{te} is achieved.

• *Different implementations*: There are different ways to treat the training instances D^{tr}. Some methods treat instances from the same seen class equally, while others treat each instance separately.

Methods treating instances from the same seen class equally. In some methods, instances from the same seen class are treated equally, and for an unseen class, instances from all seen classes are borrowed. For example, in CDAR [178], for an unseen class, instances from all seen classes are borrowed with the corresponding similarities computed based on prototypes as the weights. With these borrowed instances, a weighted SVM is learned as the classifier $f^u(\cdot)$.

In some methods, for an unseen class, only instances from some selected seen classes are borrowed. For example, in [51], for each unseen class, training instances from seen classes with high similarities (calculated based on the prototypes) are borrowed. Instances in these classes are projected to the corresponding similarity scores (called ranking scores in this work). The learned projection function is used as the classifier $f^u(\cdot)$ for the unseen classes. There are also methods that obtain borrowed instances by manually labeling similar and dissimilar seen classes for each unseen class. For example, in [145], for each unseen class, some similar seen classes and some dissimilar seen classes are labeled manually. Then, an SVM for this class is learned, considering that the prediction scores of instances from similar classes should be higher than those from dissimilar classes.

Methods treating each training instance separately. There are methods in which each instance is treated separately when borrowing instances. For example, in [57, 58], for an unseen class, ranking scores of all training instances are calculated, and top ranked instances are borrowed. Then, with the borrowed instances for all unseen classes, one-vs-rest robust SVM classifiers are learned for them.

Class-Transductive Instance-Transductive (CTIT) Setting

Under the CTIT setting, along with D^{tr}, T^s and T^u, the testing instances X^{te} are also involved in model learning. In [57, 58], besides the above introduced implementation under the CTII setting, in the process of instance borrowing for each unseen class c_i^u, testing instances can also be involved (i.e., the selected instances for class c_i^u coming from both the training instances and the testing instances). In this way, the method optimizes the parameters of the instance selection function, changes the selected instances for each unseen class and learns a robust SVM for each unseen class iteratively.

3.2.3 Synthesizing Methods. Its **insight** is to obtain labeled instances for the unseen classes by synthesizing some pseudo instances.

The **general procedure** of synthesizing methods is as follows. First, for each unseen class c_i^u, some pseudo labeled instances are synthesized. Then, with these synthesized instances for all unseen classes, classifier $f^u(\cdot)$ for the unseen classes is learned and classification of the testing instances X^{te} is achieved.

Just as with instance-borrowing methods, before synthesizing the instances, the unseen classes should be determined. Thus, methods in this category cannot be under the CIII setting. To the best of our knowledge, we only see existing methods in this category under the CTII setting, and do not see methods under the CTIT setting. Next, we introduce methods under the CTII setting.

Class-Transductive Instance-Inductive (CTII) Setting

• *Typical approach*: In the training phase, with D^{tr}, T^s and T^u, for each unseen class c_i^u, some labeled instances belonging to this class are synthesized:

$$\{\{\tilde{\mathbf{x}}_i^{U1}, c_1^u\}_{i=1}^{N_{U1}}, \{\tilde{\mathbf{x}}_i^{U2}, c_2^u\}_{i=1}^{N_{U2}}, \ldots, \{\tilde{\mathbf{x}}_i^{UN_u}, c_{N_u}^u\}_{i=1}^{N_{UN_u}}\} = g(D^{tr}, T^s, T^u), \tag{22}$$

where $\{\tilde{\mathbf{x}}_i^{Uj}, c_j^u\}$ is the i-th synthesized instance for the unseen class c_j^u, and N_{Uj} is the number of synthesized instances for the unseen class c_j^u. g is the function that synthesizes the pseudo instances. Then, with the synthesized instances, the classifier $f^u(\cdot)$ for the unseen classes $\mathcal{U}$ is learned. In the testing phase, with the learned classifier $f^u(\cdot)$, classification of the testing instances X^{te} is achieved.

• *Different implementations*: In existing methods, there are different ways to synthesize the pseudo instances and different ways to learn the classifiers.

To synthesize the pseudo instances, in some methods, instances of each class are assumed to follow some distribution. Distribution parameters for the unseen classes are firstly estimated. Then, instances of unseen classes are synthesized. For example, in [56, 144], instances in one class are assumed to follow the Gaussian distribution. In these methods, distribution parameters (the mean value and variance) of instances from each seen class c_i^s are estimated. Then, with the prototypes in the semantic space, relationships among the seen and the unseen classes are calculated. With these estimated seen class distribution parameters and calculated relationships, parameters for the distribution of unseen class instances are estimated, and pseudo instances $\{\tilde{\mathbf{x}}_j^{Ui}, c_i^u\}_{j=1}^{N_{Ui}}$ are synthesized for each unseen class c_i^u.

In some other methods, pseudo instances are synthesized through conditional data generation models. In these methods, to synthesize instances for each unseen class c_i^u, the corresponding class prototype $\mathbf{t}_i^u$ and some noises are used as the input of the generation model, to synthesize the corresponding pseudo instances $\{\tilde{\mathbf{x}}_j^{Ui}, c_i^u\}_{j=1}^{N_{Ui}}$. The noises can follow Gaussian or uniform distributions. Different models have been adopted as conditional data generation models. The generative moment matching network (GMMN) [11], denoising autoencoder [11], adversarial autoencoder [11], conditional variational autoencoder based architecture [141], and the generative adversarial networks (GANs) [11, 156, 179] are examples.

For the classifier $f^u(\cdot)$ of the unseen classes, in some works, it is in the form of zero-shot learning model with the unseen class prototypes T^u. The zero-shot learning model is learned with these synthesized instances (e.g., [56] and RKT_AO [144]) or together with the training instances D^{tr} (e.g., [156] and RKT [144]). Together with the prototypes T^u, the classifier $f^u(\cdot)$ for unseen classes can be obtained. In many other works [11, 56, 156, 179], the classifier $f^u(\cdot)$ is the standard supervised classifier, such as SVM [56] and softmax classifier [156], learned with the synthesized instances.

There are also works that synthesize pseudo instances according to the characteristics of the problem. For example, in [7], the problem is activity recognition in images. In this work, for each unseen class c_i^u, some clipart images which can embody the significant characters of this class are created manually. Classifiers (linear SVMs) are learned from these clipart images.

3.3 Discussions

In this section, we firstly give a comparison of the performance of different methods. Then, we analyze the advantages and disadvantages of methods in each category. After that, we introduce methods under special data settings. Then, we introduce multi-label zero-shot learning. Finally, we introduce generalized zero-shot learning, which is an extension of zero-shot learning.

3.3.1 Performance Comparison. Existing zero-shot learning methods are designed for problems in different areas. Thus, they are usually evaluated under different experimental settings [136]. In [155], Xian et al. compared a significant number of methods under a unified experimental setting.

Table 2. Advantages and Disadvantages of Zero-Shot Learning Methods in Each Category

Method Category		Advantages	Disadvantages
Classifier-based methods	Correspondence methods	The correspondence between the classifiers and the prototypes is captured through the correspondence function. Such a correspondence function design is simple and efficient.	They do not explicitly model the relationships among different classes, which can be useful for zero-shot learning.
	Relationship methods	The relationships among classes are modeled. In some methods, the classifiers for the seen classes learned in other problems can be directly utilized. In this way, the cost of model learning is reduced.	The relationships among classes in the semantic space is directly transferred to the feature space. The adaption problem from the semantic space to the feature space is hard to solve.
	Combination methods	If there are classifiers for the attributes learned in other problems, they can be used directly. Similar to relationship methods, the cost of model learning is reduced.	The two steps of attribute classifier learning and inferring from the attribute to the class is hard to optimize in a unified process.
Instance-based methods	Projection methods	The choice of projection functions is flexible and we can choose a suitable one according to the characteristics of the problem and the dataset. Specifically, under the CTIT learning setting, various semi-supervised learning approaches can be adopted in the projection and the classification steps.	Each unseen class has just one labeled instance (the prototype). Thus, classification methods (especially under the CIII and CTII learning settings) are usually limited to the nearest neighbor classification or some similar methods.
	Instance-borrowing methods	The number of borrowed labeled instances for unseen classes is relatively large. Thus, various classification models in supervised classification can be used.	The borrowed instances are actually instances belonging to the seen classes. This causes inaccuracy when learning the unseen class classifier.
	Synthesizing methods	The number of synthesized labeled instances for the unseen classes is relatively large and various classification models in supervised classification can be used. This is similar to instance-borrowing methods.	The synthesized instances are usually assumed to follow some distributions (usually Gaussian distribution). This causes bias of the synthesized instances.

In their comparison, popular benchmark datasets are used, and zero-shot learning methods under CIII and CTIT learning settings are compared separately. From the results we can see that under each learning setting, no method significantly outperforms all other methods for all datasets. This shows that no method can achieve the best generalizable classification results on all classification tasks. Further, we compare the results in several other papers, including [16, 56, 75, 142, 147], in which the classification results of methods under the CIII, CTII and CTIT learning settings are shown. Through the comparisons, we find that the performance of methods under the CIII and CTII settings does not have much difference. However, the performance of methods under the CTIT setting is much better than that of methods under the CIII and CTII settings. This shows that, compared with the methods under the CIII setting, the methods under the CTII setting have not made the most use of the unseen class prototypes T^u. While for the CTIT setting, as it involves

effective amount of information about the the unseen classes T^u and the testing instances X^{te} during model learning, the performance of methods under this setting is generally better than methods under the other two settings.

3.3.2 Advantages and Disadvantages. Overall, classifier-based methods and instance-based methods take different approaches to solve zero-shot learning problems. Classifier-based methods aim to learn classifiers for the unseen classes directly from the available data; while instance-based methods aim to obtain labeled instances of the unseen classes and learn classifiers from them. The advantages and disadvantages of methods in each category are summarized in Table 2.

3.3.3 Methods under Special Data Settings. There also exist variants of zero-shot learning under different data settings. Next, we introduce methods that are designed specially for these different data settings.

• *Different settings of training instances.* Typically in the feature space, besides the testing instances that are needed to classify, only labeled training instances belonging to the seen classes are provided. Some methods exploit *additional training instances* to help zero-shot learning. For example, in [29, 138], unlabeled instances from auxiliary classes (i.e., neither the seen classes nor the unseen classes) are leveraged in model learning. In [87], unlabeled instances belonging to the unseen classes (these instances are not the testing instances) are utilized to improve model learning.

• *Different settings of semantic spaces.* Typically in zero-shot learning, only one semantic space is considered, and only one prototype is defined for each class in that semantic space. Some methods consider *multiple semantic spaces.* Actually, methods for the single semantic space setting can also be used under this setting, just by concatenating prototypes from different semantic spaces for each class as one prototype, or by calculating the average results from different semantic spaces [3, 4]. There are also methods designed specifically for this multiple semantic space setting. For example, in [41], Canonical Correlation Analysis (CCA) is used to coordinate prototypes in different semantic spaces and instances in the feature space. Besides the methods considering multiple semantic spaces, there are methods considering *multiple prototypes for a class in the semantic space.* Some methods [124] utilize multiple prototypes just by using the average of results from each of them. Some methods utilize the characteristics of multiple prototypes. For example, in [103, 125], prototypes of one class are assumed to follow the Gaussian distribution. The proposed methods utilize the characteristics of Gaussian distribution when learning the classifier.

3.3.4 Multi-Label Zero-Shot Learning. In the definition of zero-shot learning (Definition 1.1), we focus on the setting in which each instance belongs to one class, i.e., has one class label. However, there are some problems in which each instance can have more than one class label. This problem setting is usually referred to as the *multi-label zero-shot learning*. When learning the classifier, existing methods either treat each class label individually [14, 99, 100, 168] or exploit multi-label correlations [44, 84, 107, 126, 172].

3.3.5 Generalized Zero-Shot Learning. In zero-shot learning, classes covered by the training and the testing instances are disjoint. In practice, this problem setting is somewhat unrealistic. In many applications, not only instances belonging to the unseen classes, but also instances belonging to the seen classes can appear in the testing phase. Under the setting of *generalized zero-shot learning* [17], the testing instances can come from both the seen and the unseen classes. Due to the co-existence of the seen and the unseen classes during the testing phase, problems under this setting are more challenging. A number of zero-shot learning methods [155] have also been tested under the generalized zero-shot learning setting. However, their performance currently is not as good as that under the setting of zero-shot learning [155]. More methods that can achieve

better performance under the setting of generalized zero-shot learning are needed to support wider practical applications.

4 APPLICATIONS

Zero-shot learning has been widely used in many areas of applications.

Computer vision. In area of computer vision, zero-shot learning has been widely used in applications related to both images and videos.

Images. In image recognition problems [33, 80], problems ranging from recognizing the general classes of objects [26, 33], animals [80], to the fine-grained classes like breeds of birds [143] and breeds of flowers [108] have been studied. Besides, zero-shot learning is also used for image segmentation [93, 104] (i.e., segmenting objects of unseen classes from the images), person pose estimation [79] (i.e., estimating pose of instances belonging to unseen classes), and person re-identification [151] (i.e., recognizing persons from unseen camera views). Finally, zero-shot learning is also used for image retrieval [124, 174, 175] (i.e., ranking images from unseen classes) and domain adaptation [165, 166] (i.e., recognizing objects from unseen domains). In image recognition and image retrieval problems, the widely used datasets for general classes include AwA [80], AwA2 [155], aPascal-aYahoo [33] and ImageNet [26], datasets for fine-grained classes include CUB [143], Oxford Flower-102 [108], and SUN Attribute [118]. A review of different datasets is available in [43].

Videos. Zero-shot learning is also widely used in problems related to videos. In recognition problems, it is used to recognize videos belonging to unseen actions [48, 51, 97, 161] and videos of unseen emotion labels [160]. In retrieval problems, zero-shot event detection (also called zero-example event detection) [15, 21, 30, 153] aims to retrieve relevant videos based on the text descriptions or just by the name of the event. Zero-shot live stream retrieval aims to retrieve live stream videos [13]. Finally, zero-shot learning is used for action localization [65, 101] (i.e., localizing actions coming from the unseen classes), event recounting [49] (i.e., recounting unseen events) and description generation [54] (i.e., generating text descriptions of unseen activities). For video recognition problems, widely used datasets include UCF101 [137], USAA [39] and HMDB51 [78]. A review of different datasets is available in [43]. For video retrieval (event detection) problems, the TRECVID dataset [112] is widely used.

Natural language processing. In the area of natural language processing (NLP), the application of zero-shot learning has been emerging in recent years. In bilingual dictionary induction [102, 105] and bilingual phrase table induction [28, 102] problems, from some seed bilingual word or phrase pairs, zero-shot learning helps to construct bilingual dictionaries or bilingual phrase tables. It is especially useful for scarce language pairs, in which word or phrase pairs are rare. In machine translation problems, zero-shot learning is used for zero-shot translation (also called zero-resource translation), for language pairs that do not have parallel corpora [37, 69, 106, 177]. Besides, zero-shot learning has also been used for spoken language understanding [36, 168] (i.e., spoken language classification in new domains) and semantic utterance classification [22] (i.e., classification of new semantic classes). Finally, it has been used in other problems related to NLP, such as entity extraction from Web pages [117], fine-grained named entity typing [99], cross-lingual document retrieval [47] and relation extraction [85].

Others. Besides the above areas, zero-shot learning has also been used for applications in other areas. In the area of ubiquitous computing, it is used for human activity recognition from sensor data [20, 148, 178]. In the area of computational biology, it is used for molecular compound analysis [81], neural decoding from fMRI images [12, 113] and ECoG [12]. In the area of security and privacy,

it is used for new transmitter recognition [127]. Other problems like knowledge representation learning [157], classifying unseen tags in StackExchange [158], unseen class article classification [107] and generating unseen emoji labels for an image [14] have also witnessed zero-shot learning based solutions.

5 FUTURE DIRECTIONS

We suggest some future research directions of zero-shot learning.

Characteristics of input data. Most existing works on zero-shot learning do not consider the characteristics of the input data in different applications. With the instances and the prototypes, some general methods are proposed, which are applicable to various applications. In practice, there exist zero-shot learning scenarios in which the input data characteristics can be utilized. For example, in sensor-based activity recognition, the input data are time series sensor readings [19]. The characteristics of time series data can be leveraged. In image object recognition, in addition to features about the whole image, features about different parts of the objects can also be considered [1, 8, 32]. In problems relevant with videos, the input data are multi-modal [153]. Characteristics of multi-modal data can be utilized when learning the zero-shot learning model [153]. In this direction, more research on exploiting the characteristics of the input data is needed.

Selection of training data. In existing works on zero-shot learning, the training data are usually predetermined labeled instances in the same feature space as the testing instances and of the same semantic type of the testing instances (e.g., they are both images about animals). It is desirable to explore other ways of selecting training data.

Heterogeneous training and testing data. Training and testing instances in existing works on zero-shot learning are usually of the same data type and of the same semantic type. However, instances in these two datasets are not necessarily of the same type. There have been approaches in which training and testing instances come from classes with different semantic types. For example, in [77], the training instances are images of objects, and the testing instances are images of scenes. To go further, the data types of the training and the testing instances can also be different. In [30], the testing instances are videos, and a part of the training instances are images. In many applications, data from other data types or semantic types can be easily obtained. Thus, the heterogeneity of the training and the testing data can be exploited in the future.

Actively selecting training data. In existing works on zero-shot learning, the seen classes and the training instances are usually predetermined. If the seen classes can be actively selected, or if the training instances can be actively labeled, the burden of acquiring and labeling instances will be reduced. There have been some efforts in actively selecting seen classes [158, 159]. More explorations in actively selecting seen classes and labeling training instances are needed.

Selection and maintenance of auxiliary information. Being similar as the training data, the auxiliary information is used for classifying the testing instances. Selecting information that is more helpful for the classification is necessary. In this direction, there have been some explorations. In [59], the attribute space is used as the semantic space. Attributes that contain more information and be more predictable are selected. Further more, the auxiliary information involved is usually the semantic information (forming the semantic space). The semantic information is involved as inspired by the way of how human beings recognize the world. However, there could be other approaches to involving auxiliary information. For example, in [145], no semantic space is involved. The auxiliary information is the human defined similarity information between classes. In [40, 92], the attribute space is used as the semantic space. Apart from the attributes defined in the semantic space, they also use learned attributes from the training data. More research on involving auxiliary

information based on the characteristics of the problems is needed. In addition, the involved auxiliary information is not necessarily in form of semantic space, and can be in other forms.

After involving the auxiliary information, how to maintain the information during model learning is another important problem. In this direction, there have been some explorations. In [18], along with the zero-shot classifier, a reconstruction model is also learned. In this way, the whole model aims to maintain more semantic information that could be discarded when just learning the classifier. More research on maintaining useful auxiliary information during model learning is needed.

More realistic and application specific problem settings. Under the setting of zero-shot learning, the testing instances are assumed just come from the unseen classes. This problem setting is somewhat unrealistic. Especially under the CTIT learning setting, the testing instances are assumed to be available during model learning. Although the performance of methods under this learning setting is generally better than that under the CIII and CTII settings, it is a more restricted and unrealistic setting. In existing works, there have been some explorations on the more realistic problem settings. In *generalized zero-shot learning*, the testing instances are assumed can come from both the seen and the unseen classes. This problem setting is more realistic. However, the performance of methods under generalized zero-shot learning is currently not as good as that under zero-shot learning [155]. Thus, approaches that can improve the performance of methods under generalized zero-shot learning are needed to explore in the future. In *open set recognition* [64, 132], the fact that unseen classes (the class labels of which can be unknown) exist is considered when learning the classification model. The learned model can determine whether a testing instance belongs to the unseen classes, but cannot determine which unseen class it belongs to. As an extension of open set recognition, in [42], Fu et al. proposed the *generalized open set recognition* learning paradigm. Under this learning paradigm, the testing instances can come from the seen and the unseen classes, while the class labels in the testing phase contain a large number of other class labels not in these two label sets. Zero-shot learning techniques are applied to solve problems under this paradigm. As the assumption of the existence of a large number of unseen classes (the labels of which can be unknown) in open set recognition is realistic in many applications, more explorations of zero-shot problem settings related with open set recognition can be performed in the future.

On the other hand, according to the characteristics of the applications, many other problem settings of zero-shot learning can be explored. For example, in some applications, the number of classes needed to recognize is large. There have been some evaluations on zero-shot learning methods under the large-scale setting [129]. In the future works, more methods that are specific for the large-scale zero-shot learning are expected. Some learning techniques, such as graph convolutional networks (GCNs) [74], can be considered to be exploited for problems under this setting. In GCNs, the relationship among different classes can be represented in the graph, and this information could be used in large-scale zero-shot learning. In some applications, the training instances and semantic information may become available in an online manner. In this scenario, the ability to utilize the sequentially available data is needed. There are existing approaches [70, 73] designed for zero-shot learning with binary attribute space as the semantic space. They use online incremental learning techniques to learn new attributes and adapt existing attributes in an online manner. In some particular learning problems, such as the image semantic segmentation problems [93, 104] and the image captioning problems [140], as the labeling process is expensive and time consuming, existing datasets just cover limited number of object classes. These problems are closely related with the classification problems, and also have their own characteristics. Zero-shot learning methods that are designed specific for these problems are also expected. In general, based on the application characteristics, more application-specific settings of zero-shot learning can be explored.

Theoretical guarantee. Existing zero-shot learning methods are usually developed in a heuristic manner, without much theoretical guarantee. Although there are some analysis on the probability of correct prediction [113] and the error bound [56, 131], we do not see theoretical analysis on problems like (1) how to choose auxiliary information, (2) what information to transfer from the training instances to help the classification of the testing instances, and (3) how to suppress uncorrelated information in the process of information transfer and avoid negative transfer [115]. More theoretical analysis on these problems will benefit the development of zero-shot learning.

Combination with other learning paradigms. As a learning paradigm, zero-shot learning can be combined with other learning paradigms to solve a wider range of problems. For example, in [109], zero-shot learning is combined with *Webly supervised learning* to classify classes that just have noisy labeled images obtained from the Web. There are also some works on applying zero-shot learning methods to the paradigm of *few-shot learning* [2, 51, 66, 100, 138].

Besides the above classification problems, zero-shot learning can also be combined with machine learning approaches for other purposes. In [53], zero-shot learning is used as the prior of *active learning* to enhance the learning process. In [63], it is combined with *lifelong learning* to learn new tasks with only descriptions. In [55, 133, 167], zero-shot learning is combined with *hashing* and forms *zero-shot hashing* problems which aim to hash images of unseen classes. In [61, 71, 111], it is combined with *reinforcement learning* to better handle new tasks, new domains and new scenarios. More combinations of zero-shot learning with other learning paradigms can be explored in future research.

6 CONCLUSION

In this article, we provide a comprehensive survey of zero-shot learning. We first motivate the need for zero-shot learning with analyzing several other learning paradigms and listing several typical application scenarios in which zero-shot learning is needed. Then, we give an overview of zero-shot learning and introduce the different learning settings. We organize the existing works on zero-shot learning from three perspectives: (1) semantic spaces, which contain the semantic information that is important for zero-shot learning; (2) methods, which are different methods for solving zero-shot learning problems under different learning settings; and (3) applications, the application areas in which zero-shot learning is used. For semantic spaces, we provide a categorization of the semantic spaces in existing works, based on whether the semantic space is engineered or learned. We also provide a summary of each category. For methods, we provide a hierarchical categorization of the methods in existing works, based on whether the methods take a classifier-oriented approach or an instance-oriented approach. We also summarize the insights of methods in each category and provide a discussion on different aspects of the methods. For applications, we introduce the applications of zero-shot learning in different areas. Finally, we suggest some future research directions for zero-shot learning from different perspectives.

ACKNOWLEDGMENTS

This research is supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative; the Interdisciplinary Graduate School, Nanyang Technological University, Singapore; and the Nanyang Assistant Professorship (NAP). It is also partially supported by the NTU-PKU Joint Research Institute, a collaboration between Nanyang Technological University and Peking University that is sponsored by a donation from the Ng Teng Fong Charitable Foundation.

REFERENCES

[1] Zeynep Akata, Mateusz Malinowski, Mario Fritz, and Bernt Schiele. 2016. Multi-Cue Zero-Shot Learning with Strong Supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 59–68.

[2] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2013. Label-Embedding for Attribute-Based Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'13)*. 819–826.

[3] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2016. Label-Embedding for Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 7 (2016), 1425–1438.

[4] Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. 2015. Evaluation of Output Embeddings for Fine-Grained Image Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 2927–2936.

[5] Ziad Al-Halah and Rainer Stiefelhagen. 2015. How to Transfer? Zero-Shot Object Recognition via Hierarchical Transfer of Semantic Attributes. In *IEEE Winter Conference on Applications of Computer Vision (WACV'15)*. 837–843.

[6] Ibrahim Alabdulmohsin, Moustapha Cisse, and Xiangliang Zhang. 2016. Is Attribute-Based Zero-Shot Learning an Ill-Posed Strategy?. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'16)*. 749–760.

[7] Stanislaw Antol, C. Lawrence Zitnick, and Devi Parikh. 2014. Zero-Shot Learning via Visual Abstraction. In *European Conference on Computer Vision (ECCV'14)*. 401–416.

[8] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, and Ruslan Salakhutdinov. 2015. Predicting Deep Zero-Shot Convolutional Neural Networks Using Textual Descriptions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 4247–4255.

[9] Abhijit Bendale and Terrance Boult. 2015. Towards open world recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 1893–1902.

[10] Irving Biederman. 1987. Recognition-by-Components: A Theory of Human Image Understanding. *Psychological Review* 94, 2 (1987), 115–147.

[11] Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. 2017. Generating Visual Representations for Zero-Shot Classification. In *IEEE International Conference on Computer Vision Workshops: Transferring and Adapting Source Knowledge in Computer Vision (ICCV Workshops'17)*. 2666–2673.

[12] Carlos A. Caceres, Matthew J. Roos, Kyle M. Rupp, Griffin Milsap, Nathan E. Crone, Michael E. Wolmetz, and Christopher R. Ratto. 2017. Feature Selection Methods for Zero-Shot Learning of Neural Activity. *Frontiers in Neuroinformatics* 11 (2017), 41.

[13] Spencer Cappallo, Thomas Mensink, and Cees G.M. Snoek. 2016. Video Stream Retrieval of Unseen Queries using Semantic Memory. In *Proceedings of the British Machine Vision Conference (BMVC'16)*.

[14] Spencer Cappallo, Thomas Mensink, and Cees G. M. Snoek. 2015. Image2Emoji: Zero-shot Emoji Prediction for Visual Media. In *Proceedings of the 23rd ACM International Conference on Multimedia (MM'15)*. 1311–1314.

[15] Xiaojun Chang, Yi Yang, Alexander G. Hauptmann, Eric P. Xing, and Yao-Liang Yu. 2015. Semantic Concept Discovery for Large-Scale Zero-Shot Event Detection. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI'15)*. 2234–2240.

[16] Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. 2016. Synthesized Classifiers for Zero-Shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 5327–5336.

[17] Wei-Lun Chao, Soravit Changpinyo, Boqing Gong, and Fei Sha. 2016. An Empirical Study and Analysis of Generalized Zero-Shot Learning for Object Recognition in the Wild. In *European Conference on Computer Vision (ECCV'16)*. 52–68.

[18] Long Chen, Hanwang Zhang, Jun Xiao, Wei Liu, and Shih-Fu Chang. 2018. Zero-Shot Visual Recognition Using Semantics-Preserving Adversarial Embedding Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

[19] Heng-Tze Cheng, Martin Griss, Paul Davis, Jianguo Li, and Di You. 2013. Towards Zero-Shot Learning for Human Activity Recognition Using Semantic Attribute Sequence Model. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'13)*. 355–358.

[20] Heng-Tze Cheng, Feng-Tso Sun, Martin Griss, Paul Davis, Jianguo Li, and Di You. 2013. NuActiv: Recognizing Unseen New Activities Using Semantic Attribute-Based Learning. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys'13)*. 361–374.

[21] Jeffrey Dalton, James Allan, and Pranav Mirajkar. 2013. Zero-Shot Video Retrieval Using Content and Concepts. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*. 1857–1860.

[22] Yann N. Dauphin, Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2014. Zero-Shot Learning for Semantic Utterance Classification. In *International Conference on Learning Representations (ICLR'14)*.

[23] Oscar Day and Taghi M Khoshgoftaar. 2017. A survey on heterogeneous transfer learning. *Journal of Big Data* 4, 1 (2017), 29.

[24] Berkan Demirel, Ramazan Gokberk Cinbis, and Nazli Ikizler-Cinbis. 2017. Attributes2Classname: A Discriminative Model for Attribute-Based Unsupervised Zero-Shot Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. 1241–1250.

[25] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. 2014. Large-Scale Object Classification using Label Relation Graphs. In *European Conference on Computer Vision (ECCV'14)*. 48–64.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*. 248–255.

[27] Zhengming Ding, Ming Shao, and Yun Fu. 2017. Low-Rank Embedded Ensemble Semantic Dictionary for Zero-Shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 6005–6013.

[28] Georgiana Dinu and Marco Baroni. 2014. How to make words with vectors: Phrase generation in distributional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*. 624–633.

[29] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2015. Improving zero-shot learning by mitigating the hubness problem. In *International Conference on Learning Representations Workshops (ICLR Workshops'15)*.

[30] Mohamed Elhoseiny, Jingen Liu, Hui Cheng, Harpreet Sawhney, and Ahmed Elgammal. 2016. Zero-Shot Event Detection by Multimodal Distributional Semantic Embedding of Videos. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. 3478–3486.

[31] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. 2013. Write a Classifier: Zero-Shot Learning Using Purely Textual Descriptions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'13)*. 2584–2591.

[32] Mohamed Elhoseiny, Yizhe Zhu, Han Zhang, and Ahmed Elgammal. 2017. Link the Head to the "Beak": Zero Shot Learning From Noisy Text Description at Part Precision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 6288–6297.

[33] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing Objects by their Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*. 1778–1785.

[34] Geli Fei, Shuai Wang, and Bing Liu. 2016. Learning Cumulatively to Become More Knowledgeable. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*. 1565–1574.

[35] Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-Shot Learning of Object Categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 4 (2006), 594–611.

[36] Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefèvre. 2015. Zero-shot semantic parser for spoken language understanding. In *16th Annual Conference of the International Speech Communication Association (INTERSPEECH'15)*. 1403–1407.

[37] Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016. Zero-Resource Translation with Multi-Lingual Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*. 268–277.

[38] Andrea Frome, Greg S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013 (NIPS'13)*. 2121–2129.

[39] Yanwei Fu, Timothy M. Hospedales, Tao Xiang, and Shaogang Gong. 2012. Attribute Learning for Understanding Unstructured Social Activity. In *European Conference on Computer Vision (ECCV'12)*. 530–543.

[40] Yanwei Fu, Timothy M. Hospedales, Tao Xiang, and Shaogang Gong. 2014. Learning Multimodal Latent Attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 2 (2014), 303–316.

[41] Yanwei Fu, Timothy M. Hospedales, Tao Xiang, and Shaogang Gong. 2015. Transductive Multi-View Zero-Shot Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 11 (2015), 2332–2345.

[42] Yanwei Fu and Leonid Sigal. 2016. Semi-supervised Vocabulary-informed Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 5337–5346.

[43] Yanwei Fu, Tao Xiang, Yu-Gang Jiang, Xiangyang Xue, Leonid Sigal, and Shaogang Gong. 2018. Recent Advances in Zero-Shot Recognition: Toward Data-Efficient Understanding of Visual Content. *IEEE Signal Processing Magazine* 35, 1 (2018), 112–125.

[44] Yanwei Fu, Yongxin Yang, Timothy Hospedales, Tao Xiang, and Shaogang Gong. 2014. Transductive Multi-label Zero-shot Learning. In *Proceedings of the British Machine Vision Conference (BMVC'14)*.

[45] Zhenyong Fu, Tao Xiang, Elyor Kodirov, and Shaogang Gong. 2015. Zero-Shot Object Recognition by Semantic Manifold Distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 2635–2644.

[46] Zhenyong Fu, Tao Xiang, Elyor Kodirov, and Shaogang Gong. 2018. Zero-Shot Learning on Semantic Class Prototype Graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 8 (2018), 2009–2022.

[47] Ruka Funaki and Hideki Nakayama. 2015. Image-Mediated Learning for Zero-Shot Cross-Lingual Document Retrieval. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 585–590.

[48] Chuang Gan, Ming Lin, Yi Yang, Yueting Zhuang, and Alexander G. Hauptmann. 2015. Exploring Semantic Inter-Class Relationships (SIR) for Zero-Shot Action Recognition. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. 3769–3775.

[49] Chuang Gan, Chen Sun, and Ram Nevatia. 2017. DECK: Discovering Event Composition Knowledge from Web Images for Zero-Shot Event Detection and Recounting in Videos. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*. 4032–4038.

[50] Chuang Gan, Tianbao Yang, and Boqing Gong. 2016. Learning Attributes Equals Multi-Source Domain Generalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 87–97.

[51] Chuang Gan, Yi Yang, Linchao Zhu, Deli Zhao, and Yueting Zhuang. 2016. Recognizing an Action Using Its Name: A Knowledge-Based Approach. *International Journal of Computer Vision* 120, 1 (2016), 61–77.

[52] Lianli Gao, Jingkuan Song, Junming Shao, Xiaofeng Zhu, and Heng Tao Shen. 2015. Zero-shot Image Categorization by Image Correlation Exploration. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval (ICMR'15)*. 487–490.

[53] Efstratios Gavves, Thomas Mensink, Tatiana Tommasi, Cees G. M. Snoek, and Tinne Tuytelaars. 2015. Active Transfer Learning With Zero-Shot Priors: Reusing Past Datasets for Future Tasks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 2731–2739.

[54] Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2013. Youtube2Text: Recognizing and Describing Arbitrary Activities Using Semantic Hierarchies and Zero-shot Recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'13)*. 2712–2719.

[55] Yuchen Guo, Guiguang Ding, Jungong Han, and Yue Gao. 2017. SitNet: Discrete Similarity Transfer Network for Zero-shot Hashing. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI'17)*. 1767–1773.

[56] Yuchen Guo, Guiguang Ding, Jungong Han, and Yue Gao. 2017. Synthesizing Samples for Zero-shot Learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI'17)*. 1774–1780.

[57] Yuchen Guo, Guiguang Ding, Jungong Han, and Yue Gao. 2017. Zero-Shot Learning With Transferred Samples. *IEEE Transactions on Image Processing* 26, 7 (2017), 3277–3290.

[58] Yuchen Guo, Guiguang Ding, Jungong Han, and Yue Gao. 2017. Zero-Shot Recognition via Direct Classifier Learning with Transferred Samples and Pseudo Labels. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*. 4061–4067.

[59] Yuchen Guo, Guiguang Ding, Jungong Han, and Sheng Tang. 2018. Zero-Shot Learning with Attribute Selection. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI'18)*.

[60] Yuchen Guo, Guiguang Ding, Xiaoming Jin, and Jianmin Wang. 2016. Transductive Zero-Shot Recognition via Shared Model Space Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. 3494–3500.

[61] Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. 2017. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*. 1480–1490.

[62] Sheng Huang, Mohamed Elhoseiny, Ahmed Elgammal, and Dan Yang. 2015. Learning Hypergraph-regularized Attribute Predictors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 409–417.

[63] David Isele, Mohammad Rostami, and Eric Eaton. 2016. Using Task Features for Zero-Shot Knowledge Transfer in Lifelong Learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. 1620–1626.

[64] Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. 2014. Multi-class Open Set Recognition Using Probability of Inclusion. In *European Conference on Computer Vision (ECCV'14)*. 393–409.

[65] Mihir Jain, Jan C. van Gemert, Thomas Mensink, and Cees G.M. Snoek. 2015. Objects2action: Classifying and localizing actions without any video example. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 4588–4596.

[66] Dinesh Jayaraman and Kristen Grauman. 2014. Zero-Shot Recognition with Unreliable Attributes. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014 (NIPS'14)*. 3464–3472.

[67] Dinesh Jayaraman, Fei Sha, and Kristen Grauman. 2014. Decorrelating Semantic Visual Attributes by Resisting the Urge to Share. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'14)*. 1629–1636.

[68] Huajie Jiang, Ruiping Wang, Shiguang Shan, Yi Yang, and Xilin Chen. 2017. Learning Discriminative Latent Attributes for Zero-Shot Classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. 4233–4242.

[69] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics* 5 (2017), 339–351.

[70] Pichai Kankuekul, Aram Kawewong, Sirinart Tangruamsub, and Osamu Hasegawa. 2012. Online Incremental Attribute-based Zero-shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12)*. 3657–3664.

[71] Ken Kansky, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. 2017. Schema Networks: Zero-shot Transfer with a Generative Causal Model of Intuitive Physics. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*. 1809–1818.

[72] Nour Karessli, Zeynep Akata, Bernt Schiele, and Andreas Bulling. 2017. Gaze Embeddings for Zero-Shot Image Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 6412–6421.

[73] Aram Kawewong, Sirinart Tangruamsub, Pichai Kankuekul, and Osamu Hasegawa. 2011. Fast Online Incremental Transfer Learning for Unseen Object Classification Using Self-Organizing Incremental Neural Networks. In *The 2011 International Joint Conference on Neural Networks (IJCNN'11)*. 749–756.

[74] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR'17)*.

[75] Elyor Kodirov, Tao Xiang, Zhenyong Fu, and Shaogang Gong. 2015. Unsupervised Domain Adaptation for Zero-Shot Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 2452–2460.

[76] Elyor Kodirov, Tao Xiang, and Shaogang Gong. 2017. Semantic Autoencoder for Zero-Shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 4447–4456.

[77] Svetlana Kordumova, Thomas Mensink, and Cees G. M. Snoek. 2016. Pooling Objects for Recognizing Scenes without Examples. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR'16)*. 143–150.

[78] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. 2011. HMDB: A Large Video Database for Human Motion Recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'11)*. 2556–2563.

[79] Alina Kuznetsova, Sung Ju Hwang, Bodo Rosenhahn, and Leonid Sigal. 2016. Exploiting View-Specific Appearance Similarities Across Classes for Zero-Shot Pose Prediction: A Metric Learning Approach. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. 3523–3529.

[80] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. 2009. Learning to Detect Unseen Object Classes by Between-Class Attribute Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*. 951–958.

[81] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data Learning of New Tasks. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI'08)*. 646–651.

[82] Angeliki Lazaridou, Elia Bruni, and Marco Baroni. 2014. Is this a wampimuk? Cross-modal mapping between distributional semantics and the visual world. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*. 1403–1414.

[83] Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. 2015. Hubness and Pollution: Delving into Cross-Space Mapping for Zero-Shot Learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL'15)*. 270–280.

[84] Chung-Wei Lee, Wei Fang, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. 2018. Multi-Label Zero-Shot Learning with Structured Knowledge Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

[85] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL'17)*. 333–342.

[86] Hanhui Li, Donghui Li, and Xiaonan Luo. 2014. BAP: Bimodal Attribute Prediction for Zero-Shot Image Categorization. In *Proceedings of the ACM International Conference on Multimedia (MM'14)*. 1013–1016.

[87] Xin Li and Yuhong Guo. 2015. Max-Margin Zero-Shot Learning for Multi-class Classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS'15)*. 626–634.

[88] Xin Li, Yuhong Guo, and Dale Schuurmans. 2015. Semi-Supervised Zero-Shot Classification with Label Representation Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 4211–4219.

[89] Xirong Li, Shuai Liao, Weiyu Lan, Xiaoyong Du, and Gang Yang. 2015. Zero-shot Image Tagging by Hierarchical Semantic Embedding. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. 879–882.

[90] Yan Li, Zhen Jia, Junge Zhang, Kaiqi Huang, and Tieniu Tan. 2018. Deep Semantic Structural Constraints for Zero-Shot Learning. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI'18)*.

[91] Yanan Li, Donghui Wang, Huanhang Hu, Yuetan Lin, and Yueting Zhuang. 2017. Zero-Shot Recognition Using Dual Visual-Semantic Mapping Paths. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 5207–5215.

[92] Yan Li, Junge Zhang, Jianguo Zhang, and Kaiqi Huang. 2018. Discriminative Learning of Latent Features for Zero-Shot Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

[93] Zhenyang Li, Efstratios Gavves, Thomas Mensink, and Cees G.M. Snoek. 2014. Attributes make sense on segmented objects. In *European Conference on Computer Vision (ECCV'14)*. 350–365.

[94] Zechao Li and Jinhui Tang. 2017. Weakly Supervised Deep Matrix Factorization for Social Image Understanding. *IEEE Transactions on Image Processing* 26, 1 (2017), 276–288.

[95] Zechao Li, Jinhui Tang, and Xiaofei He. 2018. Robust Structured Nonnegative Matrix Factorization for Image Representation. *IEEE Transactions on Neural Networks and Learning Systems* 29, 5 (2018), 1947–1960.

[96] Kongming Liang, Hong Chang, Shiguang Shan, and Xilin Chen. 2015. A Unified Multiplicative Framework for Attribute Learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 2506–2514.

[97] Jingen Liu, Benjamin Kuipers, and Silvio Savarese. 2011. Recognizing Human Actions by Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 3337–3344.

[98] Yang Long, Li Liu, Ling Shao, Fumin Shen, Guiguang Ding, and Jungong Han. 2017. From Zero-Shot Learning to Conventional Supervised Classification: Unseen Visual Data Synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 6165–6174.

[99] Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label Embedding for Zero-shot Fine-grained Named Entity Typing. In *26th International Conference on Computational Linguistics (COLING'16)*. 171–180.

[100] Thomas Mensink, Efstratios Gavves, and Cees G.M. Snoek. 2014. Costa: Co-Occurrence Statistics for Zero-Shot Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'14)*. 2441–2448.

[101] Pascal Mettes and Cees G. M. Snoek. 2017. Spatial-Aware Object Embeddings for Zero-Shot Localization and Classification of Actions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. 4453–4462.

[102] Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting Similarities among Languages for Machine Translation. *arXiv preprint arXiv:1309.4168* (2013).

[103] Tanmoy Mukherjee and Timothy Hospedales. 2016. Gaussian Visual-Linguistic Embedding for Zero-Shot Recognition. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*. 912–918.

[104] Shujon Naha and Yang Wang. 2016. Object Figure-Ground Segmentation Using Zero-Shot Learning. In *23rd International Conference on Pattern Recognition (ICPR'16)*. 2842–2847.

[105] Ndapandula Nakashole and Raphael Flauger. 2017. Knowledge Distillation for Bilingual Dictionary Induction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP'17)*. 2497–2506.

[106] Hideki Nakayama and Noriki Nishida. 2017. Zero-resource machine translation by multimodal encoder-decoder network with multimedia pivot. *Machine Translation* 31, 1-2 (2017), 49–64.

[107] Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J. Kim, and Johannes Fürnkranz. 2015. Predicting Unseen Labels Using Label Hierarchies in Large-Scale Multi-label Learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'15)*. 102–118.

[108] Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. In *Sixth Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP'08)*. 722–729.

[109] Li Niu, Ashok Veeraraghavan, and Ashutosh Sabharwal. 2018. Webly Supervised Learning Meets Zero-Shot Learning: A Hybrid Approach for Fine-Grained Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

[110] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S. Corrado, and Jeffrey Dean. 2014. Zero-Shot Learning by Convex Combination of Semantic Embeddings. In *International Conference on Learning Representations (ICLR'14)*.

[111] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. 2017. Zero-Shot Task Generalization with Multi-Task Deep Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*. 2661–2670.

[112] Paul Over, Jon Fiscus, Greg Sanders, David Joy, Martial Michel, George Awad, Alan F. Smeaton, Wessel Kraaij, and Georges Quénot. 2013. TRECVID 2013 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms, and Metrics. In *TREC Video Retrieval Evaluation Workshop (TRECVID Workshop'13)*.

[113] Mark Palatucci, Dean Pomerleau, Geoffrey Hinton, and Tom M. Mitchell. 2009. Zero-Shot Learning with Semantic Output Codes. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009 (NIPS'09)*. 1410–1418.

[114] Sinno Jialin Pan. 2014. Transfer Learning. In *Data Classification: Algorithms and Applications*. 537–570.

[115] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.
[116] Devi Parikh and Kristen Grauman. 2011. Relative Attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'11)*. 503–510.
[117] Panupong Pasupat and Percy Liang. 2014. Zero-shot Entity Extraction from Web Pages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*. 391–401.
[118] Genevieve Patterson and James Hays. 2012. SUN Attribute Database: Discovering, Annotating, and Recognizing Scene Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12)*. 2751–2758.
[119] Te Pi, Xi Li, and Zhongfei (Mark) Zhang. 2017. Boosted Zero-Shot Learning with Semantic Correlation Regularization. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI'17)*. 2599–2605.
[120] Ruizhi Qiao, Lingqiao Liu, Chunhua Shen, and Anton van den Hengel. 2016. Less is More: Zero-Shot Learning from Online Textual Documents with Noise Suppression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 2249–2257.
[121] Jie Qin, Li Liu, Ling Shao, Fumin Shen, Bingbing Ni, Jiaxin Chen, and Yunhong Wang. 2017. Zero-Shot Action Recognition with Error-Correcting Output Codes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 1042–1051.
[122] Sachin Ravi and Hugo Larochelle. 2017. Optimization as a Model for Few-Shot Learning. In *International Conference on Learning Representations (ICLR'17)*.
[123] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 5533–5542.
[124] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. 2016. Learning Deep Representations of Fine-Grained Visual Descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 49–58.
[125] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. 2016. Joint Image-Text Representation by Gaussian Visual-Semantic Embedding. In *Proceedings of the 2016 ACM Conference on Multimedia Conference (MM'16)*. 207–211.
[126] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. 2017. Multiple Instance Visual-Semantic Embedding. In *British Machine Vision Conference (BMVC'17)*.
[127] Pieter Robyns, Eduard Marin, Wim Lamotte, Peter Quax, Dave Singelée, and Bart Preneel. 2017. Physical-Layer Fingerprinting of LoRa devices using Supervised and Zero-Shot Learning. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec'17)*. 58–63.
[128] Marcus Rohrbach, Sandra Ebert, and Bernt Schiele. 2013. Transfer Learning in a Transductive Setting. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013 (NIPS'13)*. 46–54.
[129] Marcus Rohrbach, Michael Stark, and Bernt Schiele. 2011. Evaluating Knowledge Transfer and Zero-Shot Learning in a Large-Scale Setting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 1641–1648.
[130] Marcus Rohrbach, Michael Stark, György Szarvas, Iryna Gurevych, and Bernt Schiele. 2010. What Helps Where - And Why? Semantic Relatedness for Knowledge Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*. 910–917.
[131] Bernardino Romera-Paredes and Philip H.S. Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*. 2152–2161.
[132] Walter J. Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E. Boult. 2013. Toward Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 7 (2013), 1757–1772.
[133] Yuming Shen, Li Liu, Fumin Shen, and Ling Shao. 2018. Zero-Shot Sketch-Image Hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.
[134] Yutaro Shigeto, Ikumi Suzuki, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. 2015. Ridge regression, hubness, and zero-shot learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'15)*. 135–151.
[135] Richard Socher, Milind Ganjoo, Christopher D. Manning, and Andrew Y. Ng. 2013. Zero-Shot Learning Through Cross-Modal Transfer. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013 (NIPS'13)*. 935–943.
[136] Fengyi Song and Yi Chen. 2016. A survey on Zero-shot learning. *Journal of Science (JOS)* 5, 1 (2016), 455–460.
[137] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *arXiv preprint arXiv:1212.0402* (2012).
[138] Yao-Hung Hubert Tsai, Liang-Kang Huang, and Ruslan Salakhutdinov. 2017. Learning Robust Visual-Semantic Embeddings. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'17)*. 3591–3600.

[139] Naman Turakhia and Devi Parikh. 2013. Attribute Dominance: What Pops Out?. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'13)*. 1225–1232.
[140] Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2017. Captioning Images with Diverse Objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 1170–1178.
[141] Vinay Kumar Verma, Gundeep Arora, Ashish Mishra, and Piyush Rai. 2018. Generalized Zero-Shot Learning via Synthesized Examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.
[142] Vinay Kumar Verma and Piyush Rai. 2017. A Simple Exponential Family Framework for Zero-Shot Learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'17)*. 792–808.
[143] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. *The Caltech-UCSD Birds-200-2011 Dataset*. Technical Report CNS-TR-2011-001. California Institute of Technology.
[144] Donghui Wang, Yanan Li, Yuetan Lin Lin, and Yueting Zhuang. 2016. Relational Knowledge Transfer for Zero-Shot Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. 2145–2151.
[145] Gang Wang, David Forsyth, and Derek Hoiem. 2010. Comparative object similarity for improved recognition with few or no examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*. 3525–3532.
[146] Qian Wang and Ke Chen. 2017. Alternative Semantic Representations for Zero-Shot Human Action Recognition. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'17)*. 87–102.
[147] Qian Wang and Ke Chen. 2017. Zero-Shot Visual Recognition via Bidirectional Latent Embedding. *International Journal of Computer Vision* 124, 3 (2017), 356–383.
[148] Wei Wang, Chunyan Miao, and Shuji Hao. 2017. Zero-Shot Human Activity Recognition via Nonlinear Compatibility Based Method. In *Proceedings of the International Conference on Web Intelligence (WI'17)*. 322–330.
[149] Xiaoyang Wang and Qiang Ji. 2013. A Unified Probabilistic Approach Modeling Relationships between Attributes and Objects. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'13)*. 2120–2127.
[150] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. 2018. Zero-Shot Recognition via Semantic Embeddings and Knowledge Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.
[151] Zheng Wang, Ruimin Hu, Chao Liang, Yi Yu, Junjun Jiang, Mang Ye, Jun Chen, and Qingming Leng. 2016. Zero-Shot Person Re-identification via Cross-View Consistency. *IEEE Transactions on Multimedia* 18, 2 (2016), 260–272.
[152] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big Data* 3, 1 (2016), 9.
[153] Shuang Wu, Sravanthi Bondugula, Florian Luisier, Xiaodan Zhuang, and Pradeep Natarajan. 2014. Zero-shot Event Detection using Multi-modal Fusion of Weakly Supervised Concepts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'14)*. 2665–2672.
[154] Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh Nguyen, Matthias Hein, and Bernt Schiele. 2016. Latent Embeddings for Zero-Shot Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 69–77.
[155] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2017. Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly. *arXiv preprint arXiv:1707.00600v3* (2017).
[156] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. 2018. Feature Generating Networks for Zero-Shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.
[157] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Entity Descriptions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. 2659–2665.
[158] Sihong Xie, Shaoxiong Wang, and Philip S. Yu. 2016. Active Zero-Shot Learning. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM'16)*. 1889–1892.
[159] Sihong Xie and Philip S. Yu. 2017. Active zero-shot learning: a novel approach to extreme multi-labeled classification. *International Journal of Data Science and Analytics* 3, 3 (2017), 151–160.
[160] Baohan Xu, Yanwei Fu, Yu-Gang Jiang, Boyang Li, and Leonid Sigal. 2016. Video Emotion Recognition with Transferred Deep Feature Encodings. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR'16)*. 15–22.
[161] Xun Xu, Timothy Hospedales, and Shaogang Gong. 2017. Transductive Zero-Shot Action Recognition by Word-Vector Embedding. *International Journal of Computer Vision* 123, 3 (2017), 309–333.
[162] Xun Xu, Timothy M. Hospedales, and Shaogang Gong. 2016. Multi-Task Zero-Shot Action Recognition with Prioritised Data Augmentation. In *European Conference on Computer Vision (ECCV'16)*. 343–359.

[163] Xing Xu, Fumin Shen, Yang Yang, Jie Shao, and Zi Huang. 2017. Transductive Visual-Semantic Embedding for Zero-shot Learning. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (ICMR'17)*. 41–49.

[164] Xing Xu, Fumin Shen, Yang Yang, Dongxiang Zhang, Heng Tao Shen, and Jingkuan Song. 2017. Matrix Tri-Factorization with Manifold Regularizations for Zero-shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 2007–2016.

[165] Yongxin Yang and Timothy M. Hospedales. 2015. A Unified Perspective on Multi-Domain and Multi-Task Learning. In *International Conference on Learning Representations (ICLR'15)*.

[166] Yongxin Yang and Timothy M. Hospedales. 2016. Multivariate Regression on the Grassmannian for Predicting Novel Domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 5071–5080.

[167] Yang Yang, Yadan Luo, Weilun Chen, Fumin Shen, Jie Shao, and Heng Tao Shen. 2016. Zero-Shot Hashing via Transferring Supervised Knowledge. In *Proceedings of the 2016 ACM Conference on Multimedia Conference (MM'16)*. 1286–1295.

[168] Majid Yazdani and James Henderson. 2015. A Model of Zero-Shot Learning of Spoken Language Understanding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 244–249.

[169] Meng Ye and Yuhong Guo. 2017. Zero-Shot Classification with Discriminative Semantic Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 5103–5111.

[170] Felix X. Yu, Liangliang Cao, Rogerio S. Feris, John R. Smith, and Shih-Fu Chang. 2013. Designing Category-Level Attributes for Discriminative Visual Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'13)*. 771–778.

[171] Li Zhang, Tao Xiang, and Shaogang Gong. 2017. Learning a Deep Embedding Model for Zero-Shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 3010–3019.

[172] Yang Zhang, Boqing Gong, and Mubarak Shah. 2016. Fast Zero-Shot Image Tagging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 5985–5994.

[173] Ziming Zhang and Venkatesh Saligrama. 2015. Zero-Shot Learning via Semantic Similarity Embedding. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 4166–4174.

[174] Ziming Zhang and Venkatesh Saligrama. 2016. Zero-Shot Learning via Joint Latent Similarity Embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 6034–6042.

[175] Ziming Zhang and Venkatesh Saligrama. 2016. Zero-Shot Recognition via Structured Prediction. In *European Conference on Computer Vision (ECCV'16)*. 533–548.

[176] Bo Zhao, Botong Wu, Tianfu Wu, and Yizhou Wang. 2017. Zero-Shot Learning Posed as a Missing Data Problem. In *IEEE International Conference on Computer Vision Workshops: Transferring and Adapting Source Knowledge in Computer Vision (ICCV Workshops'17)*. 2616–2622.

[177] Hao Zheng, Yong Cheng, and Yang Liu. 2017. Maximum Expected Likelihood Estimation for Zero-resource Neural Machine Translation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI'17)*. 4251–4257.

[178] Vincent W. Zheng, Derek Hao Hu, and Qiang Yang. 2009. Cross-Domain Activity Recognition. In *Proceedings of the 11th International Conference on Ubiquitous Computing (UbiComp'09)*. 61–70.

[179] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. 2018. A Generative Adversarial Approach for Zero-Shot Learning From Noisy Texts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*.

Received February 2018; revised September 2018; accepted September 2018