Zero-Shot Task Adaptation with Relevant Feature Information

Atsutoshi Kumagai[1], **Tomoharu Iwata**[2], **Yasuhiro Fujiwara**[2]

[1] NTT Computer and Data Science Laboratories
[2] NTT Communication Science Laboratories
atsutoshi.kumagai@ntt.com, tomoharu.iwata@ntt.com, yasuhiro.fujiwara@ntt.com

Abstract

We propose a method to learn prediction models such as classifiers for unseen target tasks where labeled and unlabeled data are absent but a few relevant input features for solving the tasks are given. Although machine learning requires data for training, data are often difficult to collect in practice. On the other hand, for many applications, a few relevant features would be more easily obtained. Although zero-shot learning or zero-shot domain adaptation use external knowledge to adapt to unseen classes or tasks without data, relevant features have not been used in existing studies. The proposed method improves the generalization performance on the target tasks, where there are no data but a few relevant features are given, by meta-learning from labeled data in related tasks. In the meta-learning phase, it is essential to simulate test phases on target tasks where prediction model learning is required without data. To this end, our neural network-based prediction model is meta-learned such that it correctly responds to perturbations of the relevant features on randomly generated synthetic data. By this modeling, the prediction model can explicitly learn the discriminability of the relevant features without real target data. When unlabeled training data are available in the target tasks, the proposed method can incorporate such data to boost the performance in a unified framework. Our experiments show that the proposed method outperforms existing methods with four real-world datasets.

Introduction

Supervised learning requires labeled data for training. However, in real-world applications, such data are often time-consuming and impractical to collect since the label for each instance needs to be manually assigned by domain experts. Even unlabeled data might be difficult to use for training in some applications due to privacy concerns (Chen, Pastro, and Raykova 2019) and security issues (Kumagai and Iwata 2016). In contrast, *prior knowledge* about the task may be available in practice even if the data themselves are difficult to collect. In this paper, for such knowledge, we consider *relevant features* on the input feature space for solving the task. Figure 1 illustrates an example of the relevant feature information. Since full set of relevant features is generally difficult to known in advance, we assume that *a few* relevant features are available. There are many possible applications where such information is available and useful. For example, in text classification, we can associate some relevant features (words) from the names of the classes only. In medical care, even when data of patients are difficult to share due to privacy concerns, a few relevant features might be possible to share because individuals cannot be identified from them. In cyber security, although data of new attacks or malware are difficult to obtain due to scarcity, information on some of the features that characterize them is often obtainable from security reports and other sources.

Features	'python'	'kick'	'rebound'	'iphone'	'ball'	'bayes'	…	'offside'
$\mathbf{z}_{\text{soccer}}$ =	0	1	0	0	0	0	…	1
$\mathbf{z}_{\text{basketball}}$ =	0	0	1	0	0	0		0

Figure 1: An example of relevant feature information. For clarity, we consider a text classification task (soccer vs. basketball), whose feature space is a lexical space. Note that the proposed method can be applied to not only text domains. $\mathbf{z}_{\text{soccer}}$ ($\mathbf{z}_{\text{basketball}}$) is a vector representing some relevant features for the soccer (basketball) class. Here, the value 1 indicates that the corresponding feature (word) is relevant for the class and 0 means unknown. The features "kick" and "offside" are relevant features for the soccer class, while "rebound" is relevant for the basketball class. These relevant features are useful to discriminate two classes. The dimensionality of $\mathbf{z}_{\text{soccer}}$, $\mathbf{z}_{\text{basketball}}$, and feature vectors to be classified is the same.

On the other hand, even if labeled data are difficult to collect on a task of interest, called a target task, they might be available in different but related tasks, called source tasks. In the above examples, for text classification, labeled texts of other classes might be available. For medical care, data of patients in own hospital can be used. In cyber security, data of existing/old attacks or malware can be accumulated.

Zero-shot learning or zero-shot domain adaptation aims to adapt to unseen classes or tasks that never appeared during the training phase by using labeled data in other classes or tasks (Wang et al. 2019; Chen et al. 2021; Yang and Hospedales 2015b; Ishii, Takenouchi, and Sugiyama 2019). To adapt to unseen classes or tasks without data, they require auxiliary information about the classes or tasks. For

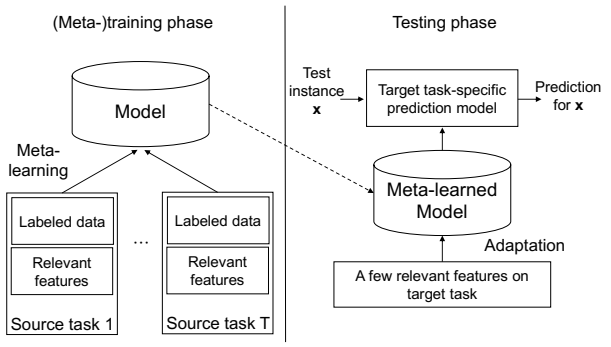


Figure 2: Our problem setting. In the (meta-)training phase, our proposed model is meta-learned with labeled data and relevant feature information in source tasks. Even when relevant features on source tasks are unavailable, they can be estimated from labeled source data by using off-the-shelf feature selection methods. In the test phase, a target task-specific prediction model is generated from the meta-learned model only with a few relevant features on the task.

example, word/sentence embeddings of classes (Frome et al. 2013; Radford et al. 2021) and hand-crafted attributes of classes (Lampert, Nickisch, and Harmeling 2013) have been used. However, existing methods do not use relevant features as auxiliary information. Although they can naively use the relevant feature vectors to condition prediction models, since such vectors are too sparse to represent classes or tasks, they are less effective as shown in our experiments.

In this paper, we propose a method to learn prediction models such as classifiers for unseen target tasks given only relevant features to solve the tasks. The proposed method is based on meta-learning, which is usually used for improving performance from a few data by using labeled data in multiple source tasks (Hospedales et al. 2020; Vanschoren 2018). In our setting, it improves the generalization performance from a few relevant features rather than data. By meta-learning, the proposed method can exploit useful information even from a few relevant features. Figure 2 shows the overview of our problem setting.

Meta-learning consists of inner and outer loop optimization problems. In the inner optimization, task-specific parameters of a model are adapted to given task-specific data. In the outer optimization, task-shared parameters are learned such that the expected test prediction performance is improved when the adapted model is used. The proposed method uses a neural network-based prediction model that consists of an embedding network and a linear model (i.e., the last layer of the whole neural network). The parameters of the embedding network and the linear model are task-shared and task-specific parameters, respectively. In the meta-learning phase, it is essential to simulate test phases on target tasks where labeled and unlabeled data are not given (Vinyals et al. 2016; Finn, Abbeel, and Levine 2017; Snell, Swersky, and Zemel 2017). To achieve this, in the inner optimization, we first generate synthetic unlabeled data from a predefined task-shared distribution. Then, the linear model is adapted such that the prediction model responds correctly to perturbations of the given task-specific relevant features on the synthetic data. Unlike simply using relevant feature vectors to condition prediction models, this formulation leads to learn prediction models that *explicitly* reflect the discriminability of the relevant features without using any real data. Although we assume that each source task has information on some relevant features, they can be estimated from labeled data using feature selection methods such as ℓ_1-regularized logistic regression (Murphy 2012). In the outer optimization, we meta-learn the parameters of the embedding network to improve the expected test prediction performance that is calculated with labeled source data. Since the solution of the inner optimization is easily calculated due to the liner model and differentiable, we can efficiently solve the outer optimization with a standard stochastic gradient descent method. By meta-learning the embedding network with various tasks, we can adapt to unseen target tasks that have only a set of few relevant features. When unlabeled data on the target tasks can be used in the inner problems, the proposed method can further incorporate such data to boost the performance in a unified framework.

Related Work

Zero-shot learning attempts to predict unseen classes, which have not appeared during the training phase (Wang et al. 2019; Chen et al. 2021; Zhao et al. 2022; Zhang, Xiang, and Gong 2017; Xu et al. 2020; Chen et al. 2022). To achieve this, it uses class descriptions such as text descriptions (Radford et al. 2021), word embedding vectors (Frome et al. 2013), and attributes (Lampert, Nickisch, and Harmeling 2013). To the best of our knowledge, there are no existing zero-shot learning methods that use relevant feature information. Note that zero-shot learning requires the class descriptions of all seen classes (in source tasks), which might be difficult to collect. In contrast, relevant features of the source tasks can be directly estimated from the given labeled data. In addition, unlike the proposed method, zero-shot learning is usually not used for regression tasks since it requires class descriptions. Zero-shot domain adaptation aims to adapt to unseen tasks by using task semantic descriptions such as the brightness of images (Yang and Hospedales 2015b), time-information (Kumagai and Iwata 2016), and device and location information (Yang and Hospedales 2015a). Unlike the proposed method, they do not use relevant features and cannot be applied to target tasks that consist of new classes. Furthermore, prediction models learned by zero-shot learning and zero-shot domain adaptation methods are usually a black box. However, in practice, the behavior of the learned model often needs to be consistent with the prior knowledge of the domain (Kerrigan, Hullman, and Bertini 2021; Beaugnon 2018). Our method explicitly trains the model so that its predictions match the prior knowledge (given relevant features).

Dataless classification aims to learns text classifiers without labeled training data by using relevant keywords (words) of the classes (Chen et al. 2015; Charoenphakdee et al. 2019; Li et al. 2018). Specifically, it uses the keywords to assign pseudo-labels to unlabeled data and then trains classifiers

with the pseudo-labeled data. However, it usually requires unlabeled data and cannot use data in source tasks.

Several studies use prior knowledge on the input feature space for regularizing models. For example, they use the similarity between features (Krupka and Tishby 2007; Mollaysa, Strasser, and Kalousis 2017; Takeishi and Kawahara 2021) and relevant features for prediction per labeled training instance (Zaidan, Eisner, and Piatko 2007; Rieger et al. 2020; Du et al. 2019). These studies have shown the effectiveness of using prior knowledge on the input feature space. However, they require labeled or unlabeled training data in target tasks and cannot treat multiple tasks.

Meta-learning aims to learn how to learn from a few data by using data in multiple source tasks (Hospedales et al. 2020; Vanschoren 2018). Although several studies use a meta-learning framework to improve the performance of zero-shot learning (Snell, Swersky, and Zemel 2017; Sung et al. 2018; Verma, Brahma, and Rai 2020; Liu et al. 2021; Cetin, Baran, and Cinbis 2022), they do not use relevant feature information as auxiliary information. Some meta-learning methods assume labeled and unlabeled data in target tasks (Ren et al. 2018; Li et al. 2022), In contrast, our method assumes a few relevant features or both a few relevant features and unlabeled data in target tasks. As a meta-learning method, the proposed method belongs to gradient-based meta-learning methods (Hospedales et al. 2020). A representative approach of gradient-based methods is model-agnostic meta-learning (MAML) (Finn, Abbeel, and Levine 2017). It solves the inner problems by using gradient descent updates of the whole neural network parameters. In contrast, several meta-learning methods update only the last layer of the neural network and have shown excellent performance in various problems (Bertinetto et al. 2019; Lee et al. 2019; Kumagai, Iwata, and Fujiwara 2021). By following them, the proposed method updates only the last layer of the neural network, which leads to efficient meta-learning.

Proposed Method

Problem Setup

We consider a binary classification task for simplicity although the proposed method can be applied to multi-class classification and regression tasks. In the (meta-)training phase, we are given T source tasks $\mathcal{D} = \{\{(\mathbf{x}_{tn}, y_{tn})\}_{n=1}^{N_t}, \mathbf{z}_t\}_{t=1}^{T}$, where T is the number of tasks, N_t is the number of data in the t-th task, $\mathbf{x}_{tn} \in \mathcal{X} \subset \mathbb{R}^D$ is the D-dimensional feature vector of the n-th instance of the t-th task, $y_{tn} \in \{c_{t0}, c_{t1}\}$ is its class label, c_{tk} is the k-th class of the t-th task, and $\mathbf{z}_t = (\mathbf{z}_{t0}, \mathbf{z}_{t1})$ is a relevant feature information vector for the t-th task. Here, $\mathbf{z}_{tk} \in \{0,1\}^D$ represents a vector of the relevant features for the k-th class in the t-th task, where the d-th element of $\mathbf{z}_{tk}$, z_{tkd}, is one if the d-th feature is relevant for the k-th class and zero if unknown. For example, in text classification tasks in Figure 1, feature space $\mathcal{X}$ is the lexical space, and words such as "kick" and "offside" are relevant features for the soccer class, while "rebound" is a relevant feature for the basketball class. We assume that no feature can be one in both classes at the same time (i.e., $\mathbf{z}_{t0}^\top \mathbf{z}_{t1} = 0$) for simplicity. Even when relevant features of source tasks $\{\mathbf{z}_t\}_{t=1}^T$ are unavailable, we can estimate them from labeled source data by using feature selection methods such as ℓ_1-regularized logistic regression[1]. In the testing phase, we are given only relevant features of an unseen target task that consists of c_0 and c_1 classes, $\mathbf{z} = (\mathbf{z}_0, \mathbf{z}_1)$, which is different from source tasks. We assume that all tasks share the same feature space $\mathcal{X}$, and the number of given relevant features, i.e., $J := \sum_{d=1}^{D}(z_{1d} + z_{0d})$, is small. Our aim is to learn an accurate classifier for the target task from only relevant features $\mathbf{z}$ by meta-learning with $\mathcal{D}$.

Model

We explain our model that outputs a task-specific classifier given a few relevant features of task $\mathbf{z}$. In the following, k-th class c_k is denoted as class k for simplicity. Our model uses the following neural network-based classifier:

$$p(y = 1|\mathbf{x}) = \sigma\left(\mathbf{w}^\top h_\theta(\mathbf{x})\right), \tag{1}$$

where $\sigma(a) = \frac{1}{1+\exp(-a)}$ is the sigmoid function, $h : \mathbb{R}^D \to \mathbb{R}^M$ is a feed-forward neural network with parameters θ for embeddings, and $\mathbf{w} \in \mathbb{R}^M$ is linear weights for classification. The parameters of neural network θ and liner weights $\mathbf{w}$ are task-shared and task-specific parameters, respectively. In this model, $\mathbf{x}$ is classified into class 1 if $\mathbf{w}^\top h_\theta(\mathbf{x}) > 0$ and class 0 otherwise. We aim to meta-learn parameters of the neural network θ to improve the expected test classification performance. The meta-learning procedure will be described in the next subsection.

We explain how to learn task-specific liner weights $\mathbf{w}$ from only $\mathbf{z}$. To this end, we assume that when the d-th feature is relevant for the class k, $z_{kd} = 1$, the following property holds: *When the d-th feature's value is increased (decreased), the k-th class's probability of the classifier is also likely to increase (decrease)*[2]. For example, in text classification, as the frequency of the words (features) such as "offside" and "hat-trick" in a text increases, the probability that the text is about the soccer (class) generally increases. In medical care, higher systolic blood pressure (feature) is known to increase the risk of the cardio-vascular disease (class) (Ma et al. 2018). Since the prior knowledge that humans can find is often simple rules, we assumed the above simple relationship between relevant features and classes. Even with such a simple assumption, we will demonstrate that the proposed method works well in our experiments.

On the basis of the above assumption, our loss function to determine $\mathbf{w}$ is designed as follows:

$$\mathcal{L}(\mathbf{w}) := \frac{1}{NJ}\sum_{n=1}^{N}\left[\sum_{d\in\mathbf{I}_n^{\mathrm{inc}}}\sigma\left(\mathbf{w}^\top h_\theta(\tilde{\mathbf{x}}_n) - \mathbf{w}^\top h_\theta(\tilde{\mathbf{x}}_n^{(d)})\right)\right.$$
$$\left. + \sum_{d\in\mathbf{I}_n^{\mathrm{dec}}}\sigma\left(\mathbf{w}^\top h_\theta(\tilde{\mathbf{x}}_n^{(d)}) - \mathbf{w}^\top h_\theta(\tilde{\mathbf{x}}_n)\right)\right] + \frac{\lambda}{2}\|\mathbf{w}\|^2, \tag{2}$$

[1]For example, features corresponding to the positive (negative) learned weights can be relevant for the positive (negative) class.

[2]If class 1's probability is expected to *decrease* as the d-th feature *increases*, the feature can be treated as relevant to class 0.

where $\{\tilde{\mathbf{x}}_n\}_{n=1}^N$ are randomly generated data from a predefined distribution p, $\tilde{\mathbf{x}}_n^{(d)} := \tilde{\mathbf{x}}_n + \delta_{nd}\mathbf{e}_d$ is the feature vector with the perturbation of the d-th feature $\delta_{nd} \in \mathbb{R}$, $\mathbf{e}_d \in \mathbb{R}^D$ is a one-hot vector whose d-th element is one, $\mathbf{I}_n^{\mathrm{inc}} := \{d \,|((z_{1d} = 1) \wedge (\delta_{nd} > 0)) \vee ((z_{0d} = 1) \wedge (\delta_{nd} < 0))\}$ is a set of feature indexes to increase class 1's probability, $\mathbf{I}_n^{\mathrm{dec}} := \{d \,|((z_{1d} = 1) \wedge (\delta_{nd} < 0)) \vee ((z_{0d} = 1) \wedge (\delta_{nd} > 0))\}$ is a set of feature indexes to decrease class 1's probability, and $\lambda > 0$ is a regularization coefficient. The specific forms of the distribution p and the perturbation δ_{nd} will be described in Section . The value of the first term in Eq. (2) decreases when the positive perturbation of class 1's relevant feature increases class 1's probability (or, class 1's score $\mathbf{w}^\top h_\theta(\mathbf{x})$), or the negative perturbation of class 0's relevant feature increases class 1's probability. This loss behavior reflects the assumption made above. Similarly, the value of the second term in Eq. (2) decreases when the probability of class 1 decreases with the perturbations of the relevant features. The third term is the regularizer to evade the overfitting. By minimizing this loss, we can find $\mathbf{w}$ that reflects the discriminability of given relevant features.

We minimize the loss in Eq. (2) by using a standard gradient descent method. Specifically, we update $\mathbf{w}$ by using the following gradient step with step size $\alpha > 0$,

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}), \tag{3}$$

where

$$\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \frac{1}{NJ} \sum_{n=1}^{N} \left[\sum_{d \in \mathbf{I}_n^{\mathrm{inc}}} \sigma(\mathbf{w}^\top \mathbf{u}_{nd})(1 - \sigma(\mathbf{w}^\top \mathbf{u}_{nd}))\mathbf{u}_{nd} - \sum_{d \in \mathbf{I}_n^{\mathrm{dec}}} \sigma(-\mathbf{w}^\top \mathbf{u}_{nd})(1 - \sigma(-\mathbf{w}^\top \mathbf{u}_{nd}))\mathbf{u}_{nd} \right] + \lambda \mathbf{w},$$

$$\mathbf{u}_{nd} := h_\theta(\tilde{\mathbf{x}}_n) - h_\theta(\tilde{\mathbf{x}}_n^{(d)}). \tag{4}$$

By updating I times with Eq. (3), we can obtain adapted weights $\mathbf{w}_*(\theta; \mathcal{S}, \mathbf{z})$ where $\mathcal{S} := \{\tilde{\mathbf{x}}_n\}_{n=1}^N$. Then the task-specific classifier is represented as

$$p(y = 1|\mathbf{x}) = \sigma(\mathbf{w}_*(\theta; \mathcal{S}, \mathbf{z})^\top h_\theta(\mathbf{x})). \tag{5}$$

Meta-learning

We explain the meta-training procedure of our model. In this section, symbol $\mathcal{S}$ is used for synthetic unlabeled data for source tasks and class labels in each source task are represented as $\{0, 1\}$. In the outer optimization, the parameters to be meta-learned are parameters of neural network θ. Our approach maximizes the expected test classification performance when the classifier obtained in the inner optimization (Section) is used:

$$\min_\theta \mathbb{E}_{t \sim \{1,\ldots,T\}} \mathbb{E}_{\tilde{\mathbf{z}} \sim \mathbf{z}_t} \mathbb{E}_{\mathcal{Q} \sim \mathcal{D}_t} \mathbb{E}_{\mathcal{S} \sim p} \left[-\ln p(\mathcal{Q}; \theta, \mathcal{S}, \tilde{\mathbf{z}}) \right], \tag{6}$$

where $\mathcal{Q} := \{(\mathbf{x}_n, y_n)\}_{n=1}^{N_\mathcal{Q}}$ are testing data (a query set) drawn from the t-th task, $\mathcal{S} := \{\tilde{\mathbf{x}}_n\}_{n=1}^N$ are synthetic unlabeled data drawn from the predefined distribution p, and

Algorithm 1: Meta-training procedure of our model.

Require: Source tasks $\mathcal{D}$, synthetic unlabeled data size N, query set size $N_\mathcal{Q}$, the number of the gradient steps I, step size α, the number of the relevant features J, and the predefined distribution for synthetic data p
Ensure: Task-shared parameters of neural network θ

1: **repeat**
2: Randomly sample task t from $\{1, \ldots, T\}$
3: Randomly sample J relevant features $\tilde{\mathbf{z}}$ from $\mathbf{z}_t$
4: Randomly sample synthetic unlabeled data $\mathcal{S}$ with size N from predefined distribution p
5: Randomly sample query set $\mathcal{Q}$ with size $N_\mathcal{Q}$ from $\mathcal{D}_t$
6: Initialize linear weights $\mathbf{w}$ as zeros
7: **for** $i := 1$ to I **do**
8: Update task-specific linear weights $\mathbf{w}$ by Eq. (3)
9: **end for**
10: Calculate the loss on $\mathcal{Q}$ by Eq. (7)
11: Update parameters θ with the gradients of the loss
12: **until** End condition is satisfied;

$\tilde{\mathbf{z}} \sim \mathbf{z}_t$ denotes that J relevant features are randomly chosen from the given relevant features of the t-th task. Log probability $\ln p(\mathcal{Q}; \theta, \mathcal{S}, \tilde{\mathbf{z}})$ is described as

$$\begin{aligned} \ln p(\mathcal{Q}; \theta, \mathcal{S}, \tilde{\mathbf{z}}) &= \sum_{n=1}^{N_\mathcal{Q}} \ln p(y_n|\mathbf{x}_n; \theta, \mathcal{S}, \tilde{\mathbf{z}}) \\ &= \sum_{n=1}^{N_\mathcal{Q}} y_n \ln \sigma\left(\mathbf{w}_*(\theta; \mathcal{S}, \tilde{\mathbf{z}})^\top h_\theta(\mathbf{x})\right) \\ &\quad + (1 - y_n) \ln\left(1 - \sigma\left(\mathbf{w}_*(\theta; \mathcal{S}, \tilde{\mathbf{z}})^\top h_\theta(\mathbf{x})\right)\right). \end{aligned} \tag{7}$$

Since adapted weights $\mathbf{w}_*(\theta; \mathcal{S}, \mathbf{z})$ are easily obtained by using the closed-form gradient steps in Eq. (4), the outer optimization problem in Eq. (6) is efficiently constructed. In addition, since $\mathbf{w}_*(\theta; \mathcal{S}, \mathbf{z})$ are differentiable, the outer problem is also differentiable. Therefore, we can solve it by using standard stochastic gradient methods such as Adam (Kingma and Ba 2014). Algorithm 1 shows the pseudocode for our meta-training procedure. For each iteration, we randomly sample task t from source tasks (Line 2). From the t-th task, we randomly sample J relevant features $\tilde{\mathbf{z}}$ (Line 3), synthetic unlabeled data $\mathcal{S}$ (Line 4), and query set $\mathcal{Q}$ (Line 5). We initialize linear weights $\mathbf{w}$ as zeros (Line 6). By repeating the gradient steps I times, we obtain the classifier adapted to a few relevant features $\tilde{\mathbf{z}}$ (Lines 7–9). Lastly, we calculate the loss on a query set $\mathcal{Q}$ (Line 10) and update task-shared parameters θ with the gradient of the loss (Line 11). After meta-learning θ, given a few relevant features $\mathbf{z}$ on the target task, we can obtain the target task-specific classifier by executing Lines 6 to 9 with $\mathbf{z}$ on Algorithm 1. Since our model is meta-learned to estimate accurate classifiers from a few relevant features on multiple source tasks, we can expect that it can generalize to the target tasks.

Adaptation with Unlabeled Data

When unlabeled data in the target task $\{\mathbf{x}_n\}_{n=1}^{N_\mathrm{U}}$ are available for training, the proposed method can incorporate them

to boost the performance. Specifically, in the inner optimization, our loss function is designed as follows:

$$\mathcal{L}_{\mathrm{U}}(\mathbf{w}) := -\frac{1}{N_{\mathrm{U}}}\sum_{n=1}^{N_{\mathrm{U}}} \ln p(\tilde{y}_n|\mathbf{x}_n;\mathbf{w}) + \mu\mathcal{L}(\mathbf{w}), \qquad (8)$$

where the first term is the average of the log probability of data $(\mathbf{x}_n, \tilde{y}_n)$, the second term $\mathcal{L}(\mathbf{w})$ is the loss in Eq. (2), and $\mu > 0$ is a regularization parameter. Here, label $\tilde{y}_n$ is assigned in advance by using information of a few relevant features. Although the proposed method can use any pseudo-labeling strategies, we simply assign label $\tilde{y}_n$ by one-nearest neighbor with class prototypes $\mathbf{z}_0$ and $\mathbf{z}_1$ in our experiments. In the meta-learning phase, we use the same loss form $\mathcal{L}_{\mathrm{U}}(\mathbf{w})$ calculated with both unlabeled data in each source task and synthetic data in the inner optimization to simulate the test phases.

Experiments

Data

We used four real-world datasets: 20News[3], WoS[4], URL[5] and Mnistr[6]. 20News consisted of 18,846 newsgroup posts on 20 topics (classes). Each post was represented by bag-of-words. We removed headers, signature blocks, and quotation blocks in posts, and omitted words that occurred in fewer than 100 posts and stop words. The number of features was 1,336. WoS consisted of 46,985 published papers with 134 classes available from the Web of Science (Kowsari et al. 2017). Each paper was represented by bag-of-words. We omitted words that occurred in fewer than 400 papers and stop words and omitted classes with fewer than 350 papers. The number of features and classes were 2,045 and 79. We note that bag-of-words features are still strong features in text classification tasks (Sato, Yamada, and Kashima 2022). In fact, recent studies have reported that logistic regression with bag-of-ward features performs comparatively to or better than recent methods such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2018) in text classification tasks (Lin et al. 2023). URL consisted of Uniform Resource Locators (URLs) of 15 classes such as "Arts", "Business", and "Adult". We separated each URL path by "/" and treated the separated elements as features. Each feature took zero or one. We randomly select 1,000 instances from each class. The number of features was 2,921. Mnist-r is created from the Mnist dataset by rotating the images (Ghifary et al. 2015). This dataset has six domains (six rotating angles) with 10 digit labels. Each class of each domain has 100 instances and its feature dimension is 256. Thus, the total number of classes was 60. Each feature was normalized in the range $[0, 1]$.

For 20News, WoS, and URL, all feature vectors are normalized by ℓ_1-normalization. For 20News, we randomly used 10 classes for training, 5 classes for validation, and 5 classes for testing. For WoS, we randomly used 69 classes for training, 5 classes for validation, and 5 classes for testing. For URL, we randomly used 7 classes for training, 4 classes for validation, and 4 classes for testing. For Mnistr, we randomly used 30 classes for training, 15 classes for validation, and 15 classes for testing. For testing, 30 binary classification tasks were created from the testing classes. Relevant features of each target task were extracted by ℓ_1-regularized logistic regression. Specifically, we selected features corresponding to the upper $\frac{J}{2}$ and lower $\frac{J}{2}$ values of the learned weights. For each dataset, we randomly created five splits of training/validation/testing classes and evaluated average test accuracy with the number of relevant features J on the target tasks being $\{2, 4\}$.

Comparison Methods

We compared the proposed method (Ours) with nine comparison methods in a setting where no target unlabeled data are given: keyword matching (KW), k-nearest neighbor classifier (kNN), logistic regression (LR), neural network-based classifier (NN), logistic regression differentiable discriminator (LRD2) (Bertinetto et al. 2019), prototypical network (Proto) (Snell, Swersky, and Zemel 2017), neural processes (NP) (Garnelo et al. 2018), relation network (Relation) (Sung et al. 2018), and model-agnostic metric learning (Metric) (Shen et al. 2020). In a setting where target unlabeled data are available, we compared the proposed method (OursU) with four extensions of the above comparison methods with unlabeled data: LRU, LRD2U, ProtoU, and NPU where the 'U' denotes using unlabeled data. KW classifies test data by comparing the number of non-zero values in the given relevant features of each class. kNN classifies test data by using one-nearest neighbor on the basis of class prototypes $\mathbf{z}_0$ and $\mathbf{z}_1$. Cosine similarity is used as the distance metric. LR, NN, and LRD2 use pseudo-labeled synthetic data on target tasks for training. To assign pseudo-labels to synthetic data, these methods use the above kNN classifier. LR and NN use a logistic regression model and a feed-forward neural network classifier model, respectively. LR and NN do not use labeled data in source tasks. LRD2 is a meta-learning method that learns how to learn from pseudo-labeled synthetic data using labeled data in source tasks. It adapts the last layer of the neural network in the inner optimization as in the proposed method. Proto, NP, Relation, and Metric are widely used meta-learning-based zero-shot learning methods that use labeled data in source tasks. Proto learns two embedding networks for instance and relevant feature vectors of classes, respectively, such that embedded data in a class are close to the embedding of the class. NP models classifiers by a feed-forward neural network that takes the concatenation of instance, and relevant feature vectors of classes as input. Relation learns neural networks that output the relation score of instance and relevant feature vector of a class. Metric learns both class and instance embeddings in the unit norm space. LRU, and LRD2U use unlabeled data instead of synthetic data for pseudo-labeling. LRD2U is equivalent to OursU without the proposed regularizer (i.e., $\mu = 0$ in Eq. (8)). ProtoU modifies the class embeddings of Proto with unlabeled data on the basis of soft

[3]http://qwone.com/ jason/20Newsgroups/

[4]https://github.com/kk7nc/HDLTex

[5]https://www.kaggle.com/datasets/shawon10/url-classification-dataset-dmoz

[6]https://github.com/ghif/mtae

Data	J	Ours	KW	kNN	LR	NN	LRD2	Proto	NP	Relation	Metric
20News	2	**62.46**	21.58	21.80	56.34	52.51	50.05	58.06	57.39	58.79	57.92
	4	**65.17**	30.92	31.47	57.87	54.00	50.08	64.23	57.68	62.61	**65.58**
WoS	2	**70.73**	57.09	57.79	60.69	53.77	50.05	60.33	64.73	60.53	59.33
	4	**77.49**	71.94	72.87	63.31	56.22	50.80	**76.33**	**75.43**	73.67	72.93
URL	2	**53.29**	1.74	1.74	50.77	49.99	50.22	51.37	51.95	49.05	51.16
	4	**53.48**	3.05	3.05	50.79	50.50	50.84	48.41	52.28	51.34	51.40
Mnistr	2	**77.23**	41.19	**74.72**	61.19	56.27	57.77	73.93	67.51	72.47	**75.43**
	4	83.25	55.53	**85.92**	67.22	60.64	63.80	81.39	73.15	81.17	83.49
Average		**67.89**	35.38	43.67	58.53	54.24	52.95	64.26	62.51	63.70	64.66

Table 1: Average test accuracy [%] on each dataset in the setting where unlabeled target data are unavailable. Boldface denotes the best and comparable methods according to the paired t-test ($p = 0.05$).

k-means (Ren et al. 2018). NPU modifies the output of NP using unlabeled data. In the inner optimization, the last layer of the neural network is adapted to unlabeled data by performing the entropy minimization (Wang et al. 2021).

Settings

Synthetic Data For the proposed method, LR, NN, and LRD2, the synthetic instance was generated from p as follows: First, each feature was uniformly randomly generated from $[0, 1)$. Second, 70% features of the generated instance were randomly masked with zero since original datasets were sparse. Last, except for Mnistr, the generated data were normalized by ℓ_1-normalization to match the preprocessing described in Section .

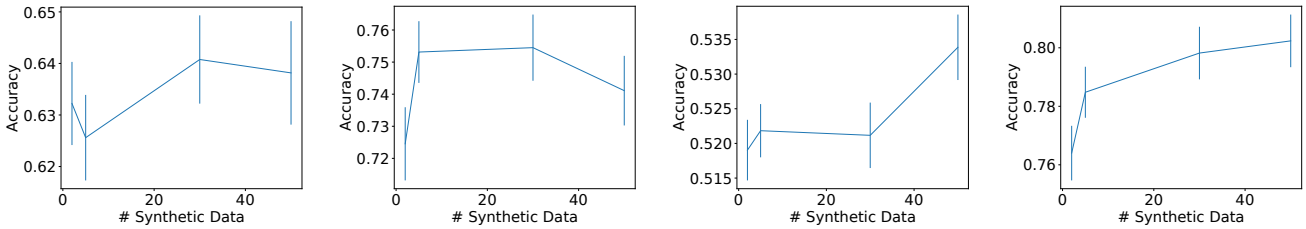
Perturbation For the proposed method, perturbation $\delta_{nd} \in \mathbb{R}$ in Eq. (2) was deterministically generated as follows: When the original feature value $\tilde{x}_{nd} < 0.5$, $\delta_{nd} = 1 - \tilde{x}_{nd}$. When $\tilde{x}_{nd} \geq 0.5$, $\delta_{nd} = -\tilde{x}_{nd}$. This perturbation maximally changes the original feature value and the perturbed feature $x_{nd}^{(d)}$ takes zero or one. Although this perturbation strategy worked well in our experiments, other strategies such as random perturbations are of course possible.

Network Architecture For the proposed method, LRD2, LRD2U, Proto, ProtoU, and Metric, a three-layered feed-forward neural network with 128 hidden and output nodes and ReLU activation was used for an instance embedding network h. For Proto, ProtoU, and Metric, the same neural network architecture was used for the embedding network of the relevant feature vectors. For NN, NP, and NPU, one classification layer was added to the embedding network. For Relation, two two-layered feed-forward neural networks were used for embeddings of the instance and relevant feature vector, respectively. Then, a three-layered feed-forward neural network was used for outputting relation scores from the embeddings.

Hyperparameters For LR and LRU, the regularization parameter was chosen from $\{10^{-3}, 10^{-2}, \ldots, 10^{5}\}$. For the proposed method, LR, NN, and LRD2, the number of synthetic unlabeled data N was 50. For the proposed method, LRD2, LRD2U, and NPU, the step size of gradient descent α and the iteration number I in the inner problems were selected from $\{10, 1, 10^{-1}\}$ and $\{2, 5, 10\}$, respectively. For the proposed method, regularization parameters λ and μ were selected from $\{1, 10^{-1}, 10^{-2}, 0\}$ and $\{10, 1, 10^{-1}, 10^{-2}, 10^{-3}\}$, respectively. When using OursU, λ was set to zero. For the proposed method, LRD2, LRD2U, Proto, ProtoU, NP, NPU, Relation, and Metric, relevant features of each source task were created by ℓ_1-regularized logistic regression. Specifically, we selected features corresponding to the upper 10 and lower 10 values of the learned weights. For all neural network-based methods, we used the Adam optimizer with a learning rate of 10^{-3} (Kingma and Ba 2014). The validation accuracy was also used for early stopping to avoid overfitting, where the maximum number of training iterations was set to $10,000$. Query set size $N_{\mathcal{Q}}$ was set to 100. All neural network-based methods were implemented using Pytorch (Paszke et al. 2017). For the methods that do not use data in source tasks (KW, kNN, LR, LRU, and NN), the best test results were reported from the hyperparameter candidates. For the other methods including the proposed method, the hyperparameters were determined on the basis of the mean validation accuracy. All experiments were conducted on a Linux server with A100 GPU and 2.20Hz Intel Xeon CPU.

Results

Table 1 shows the average test accuracies in the setting where unlabeled training data are unavailable in the target tasks. The proposed method performed the best or comparably well in 7 out of 8 cases. KW and kNN, which use only relevant features to classify test data, performed worse in all cases except for kNN with Mnistr. These results indicate that relevant features alone are generally insufficient for classification. Similarly, LR, NN, and LRD2, which use pseudo-labeled synthetic data for training classifiers, did not perform well. Since synthetic data do not have the information of real data distribution, it was difficult to learn classifiers that can distinguish real data. In contrast, since our proposed loss function in Eq. (2) does not explicitly use the information of the data distribution, it worked well even with synthetic data. Zero-shot learning methods (Proto, NP, Relation, and Metric) performed better than other existing methods by directly inferring classifiers from relevant feature vectors. However, they performed worse than the proposed method. This result suggests that it is insufficient to naively apply the relevant feature vectors to zero-shot learn-



(a) 20News (b) WoS (c) URL (d) Mnistr

Figure 3: Average test accuracies and their standard errors of Ours when changing the number of synthetic data.

ing methods, and our framework can exploit more useful information from a few relevant features by explicitly guiding the prediction outputs. For Mnistr, the proposed method performed worse than kNN when $J = 4$. This result indicates that useful knowledge on source tasks was difficult to transfer to target ones in Mnistr. However, as shown in Table 3 below, the proposed method can work much better than kNN when unlabeled training data can be used for training.

Figure 3 shows the average test accuracies of the proposed method when changing the number of synthetic data. As the number of synthetic data increased, the proposed method tended to perform better. This is because the classifier can be learned from given relevant features at many points in the feature space when the number of synthetic data is large.

We conducted an ablation study of our meta-learning design. We evaluated three variants of the proposed method: Target, Ladapt, and Uadapt. Target is our method without meta-learning on source tasks. This method learns the whole neural network classifier by minimizing Eq. (2) with given target relevant features. Ladapt is our method with adaptation to labeled data in the inner optimization of the meta-learning phase. Specifically, this method adapts the last layer of the neural network by minimizing the binary cross entropy loss on labeled data. Uadapt is our method with adaptation to unlabeled data instead of synthetic data in the inner optimization. Table 2 shows the results. The proposed method outperformed the other methods. Target performed worse than the proposed method in most cases, which indicates the effectiveness of meta-learning. Ladapt also did not work well even though it uses information of source tasks. This is because the loss used in the meta-learning phase was different to that (Eq. (2)) used in the test phase, and thus, the embedding network is not meta-learned to generalize from given relevant features. Uadapt was slightly worse than the proposed method. This is because the distributions of real unlabeled and synthetic data are different, and thus, the test phases were slightly difficult to simulate. As a result, these results show the validity of our meta-learning design.

Table 3 shows the average test accuracies in the setting where unlabeled training data are available in the target tasks. All methods tended to improve the performance by using additional information of unlabeled data. The proposed method performed the best or comparably well in most cases. Especially, the proposed method outperformed LRD2U. Since the difference between both methods is whether or not our proposed loss in Eq . (2) was used, this result indicates the effectiveness of our proposed loss function when unlabeled training data are available.

Data	J	Ours	Target	Ladapt	Uadapt
20News	2	**62.46**	53.05	50.01	61.01
	4	**65.17**	53.72	50.05	62.83
WoS	2	**70.73**	52.11	51.69	**69.52**
	4	**77.49**	54.62	56.37	**76.46**
URL	2	**53.29**	50.47	50.00	51.52
	4	**53.48**	50.61	50.67	**53.36**
Mnistr	2	**77.23**	73.47	50.89	64.43
	4	**83.25**	81.77	51.02	72.00
Average		**67.89**	58.73	51.34	63.89

Table 2: Ablation study: average test accuracy [%] in the setting where unlabeled target data are unavailable.

Data	J	OursU	LRU	LRD2U	ProtoU	NPU
20	2	**65.03**	55.63	61.33	58.03	58.57
News	4	**69.28**	58.42	65.09	61.75	56.50
WoS	2	73.31	71.77	**76.88**	68.95	55.54
	4	**83.92**	82.63	76.67	72.77	66.05
URL	2	**52.20**	50.01	49.63	51.41	**53.47**
	4	**51.02**	50.09	49.65	49.45	**52.00**
Mnistr	2	**83.10**	80.99	82.23	**81.99**	49.54
	4	**87.94**	**87.15**	**87.77**	**87.33**	67.77
Avg.		**70.73**	67.08	68.68	66.46	57.44

Table 3: Average test accuracy [%] with unlabeled target data on each dataset. The numbers of unlabeled target and synthetic data were 20 and 50, respectively.

Conclusion

In this paper, we proposed a method to learn prediction models for unseen target tasks that have only information on a few relevant features for the tasks. By using the meta-learning framework, the proposed method learns how to learn from a few relevant features on multiple source tasks. In addition, when unlabeled training data are available, the proposed method can use them to boost the performance. Our experiments showed that the proposed method performed well in both cases where unlabeled training data are unavailable/available in target tasks with four real-world datasets. For future work, we will extend our framework to treat prior knowledge that describes more complex relationships between features and labels.

References

Beaugnon, A. 2018. *Expert-in-the-loop supervised learning for computer security detection systems*. Ph.D. thesis, Université Paris sciences et lettres.

Bertinetto, L.; Henriques, J. F.; Torr, P. H.; and Vedaldi, A. 2019. Meta-learning with differentiable closed-form solvers. In *ICLR*.

Cetin, S.; Baran, O. B.; and Cinbis, R. G. 2022. Closed-form sample probing for learning generative models in zero-shot Learning. In *ICLR*.

Charoenphakdee, N.; Lee, J.; Jin, Y.; Wanvarie, D.; and Sugiyama, M. 2019. Learning only from relevant keywords and unlabeled documents. In *EMNLP-IJCNLP*.

Chen, J.; Geng, Y.; Chen, Z.; Horrocks, I.; Pan, J. Z.; and Chen, H. 2021. Knowledge-aware zero-shot learning: Survey and perspective. In *IJCAI*.

Chen, S.; Hong, Z.; Liu, Y.; Xie, G.-S.; Sun, B.; Li, H.; Peng, Q.; Lu, K.; and You, X. 2022. Transzero: attribute-guided transformer for zero-shot learning. In *AAAI*.

Chen, V.; Pastro, V.; and Raykova, M. 2019. Secure computation for machine learning with SPDZ. *arXiv preprint arXiv:1901.00329*.

Chen, X.; Xia, Y.; Jin, P.; and Carroll, J. 2015. Dataless text classification with descriptive LDA. In *AAAI*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Du, M.; Liu, N.; Yang, F.; and Hu, X. 2019. Learning credible deep neural networks with rationale regularization. In *ICDM*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-Learning for fast adaptation of deep networks. In *ICML*.

Frome, A.; Corrado, G. S.; Shlens, J.; Bengio, S.; Dean, J.; Ranzato, M.; and Mikolov, T. 2013. Devise: a deep visual-semantic embedding model. *NeurIPS*.

Garnelo, M.; Rosenbaum, D.; Maddison, C.; Ramalho, T.; Saxton, D.; Shanahan, M.; Teh, Y. W.; Rezende, D.; and Eslami, S. A. 2018. Conditional neural processes. In *ICML*.

Ghifary, M.; Kleijn, W. B.; Zhang, M.; and Balduzzi, D. 2015. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*.

Hospedales, T.; Antoniou, A.; Micaelli, P.; and Storkey, A. 2020. Meta-learning in neural networks: a survey. *arXiv preprint arXiv:2004.05439*.

Ishii, M.; Takenouchi, T.; and Sugiyama, M. 2019. Zero-shot domain adaptation based on attribute information. In *ACML*.

Kerrigan, D.; Hullman, J.; and Bertini, E. 2021. A survey of domain knowledge elicitation in applied machine learning. *Multimodal Technologies and Interaction*, 5(12): 73.

Kingma, D. P.; and Ba, J. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kowsari, K.; Brown, D. E.; Heidarysafa, M.; Meimandi, K. J.; Gerber, M. S.; and Barnes, L. E. 2017. Hdltex: hierarchical deep learning for text classification. In *ICMLA*.

Krupka, E.; and Tishby, N. 2007. Incorporating prior knowledge on features into learning. In *AISTATS*.

Kumagai, A.; and Iwata, T. 2016. Learning future classifiers without additional data. In *AAAI*.

Kumagai, A.; Iwata, T.; and Fujiwara, Y. 2021. Meta-learning for relative density-ratio estimation. In *NeurIPS*.

Lampert, C. H.; Nickisch, H.; and Harmeling, S. 2013. Attribute-based classification for zero-shot visual object categorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(3): 453–465.

Lee, K.; Maji, S.; Ravichandran, A.; and Soatto, S. 2019. Meta-learning with differentiable convex optimization. In *CVPR*.

Li, C.; Kothawade, S.; Chen, F.; and Iyer, R. 2022. Platinum: semi-supervised model agnostic meta-learning using submodular mutual information. In *ICML*.

Li, X.; Li, C.; Chi, J.; Ouyang, J.; and Li, C. 2018. Dataless text classification: a topic modeling approach with document manifold. In *CIKM*.

Lin, Y.-C.; Chen, S.-A.; Liu, J.-J.; and Lin, C.-J. 2023. Linear classifier: an often-forgotten baseline for text classification. In *ACL*.

Liu, Z.; Li, Y.; Yao, L.; Wang, X.; and Long, G. 2021. Task aligned generative meta-learning for zero-shot learning. In *AAAI*.

Ma, F.; Gao, J.; Suo, Q.; You, Q.; Zhou, J.; and Zhang, A. 2018. Risk prediction on electronic health records with prior medical knowledge. In *KDD*.

Mollaysa, A.; Strasser, P.; and Kalousis, A. 2017. Regularising non-linear models using feature side-information. In *ICML*.

Murphy, K. P. 2012. *Machine learning: a probabilistic perspective*. MIT press.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J. B.; Larochelle, H.; and Zemel, R. S. 2018. Meta-learning for semi-supervised few-shot classification. In *ICLR*.

Rieger, L.; Singh, C.; Murdoch, W.; and Yu, B. 2020. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. In *ICML*.

Sato, R.; Yamada, M.; and Kashima, H. 2022. Re-evaluating word movers distance. In *ICML*.

Shen, J.; Wang, H.; Zhang, A.; Qiu, Q.; Zhen, X.; and Cao, X. 2020. Model-agnostic metric for zero-shot learning. In *WCCV*.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.

Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H.; and Hospedales, T. M. 2018. Learning to compare: relation network for few-shot learning. In *CVPR*.

Takeishi, N.; and Kawahara, Y. 2021. Knowledge-based regularization in generative modeling. In *IJCAI*.

Vanschoren, J. 2018. Meta-learning: a survey. *arXiv preprint arXiv:1810.03548*.

Verma, V. K.; Brahma, D.; and Rai, P. 2020. Meta-learning for generalized zero-shot learning. In *AAAI*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *NeurIPS*.

Wang, D.; Shelhamer, E.; Liu, S.; Olshausen, B.; and Darrell, T. 2021. Tent: fully test-time adaptation by entropy minimization. In *ICLR*.

Wang, W.; Zheng, V. W.; Yu, H.; and Miao, C. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2): 1–37.

Xu, W.; Xian, Y.; Wang, J.; Schiele, B.; and Akata, Z. 2020. Attribute prototype network for zero-shot learning. In *NeurIPS*.

Yang, Y.; and Hospedales, T. 2015a. A unified perspective on multi-domain and multi-task learning. In *ICLR*.

Yang, Y.; and Hospedales, T. 2015b. Zero-shot domain adaptation via kernel regression on the grassmannian. *arXiv preprint arXiv:1507.07830*.

Zaidan, O.; Eisner, J.; and Piatko, C. 2007. Using annotator rationales to improve machine learning for text categorization. In *NAACL*.

Zhang, L.; Xiang, T.; and Gong, S. 2017. Learning a deep embedding model for zero-shot learning. In *CVPR*.

Zhao, X.; Shen, Y.; Wang, S.; and Zhang, H. 2022. Boosting generative zero-shot learning by synthesizing diverse features with attribute augmentation. In *AAAI*.