TART: Improved Few-shot Text Classification Using Task-Adaptive Reference Transformation

Shuo Lei[†], **Xuchao Zhang**[‡], **Jianfeng He**[†], **Fanglan Chen**[†], **Chang-Tien Lu**[†]
[†]Department of Computer Science, Virginia Tech, Falls Church, VA, USA
[‡]Microsoft, Redmond, WA, USA
{slei,jianfenghe,fanglanc,ctlu}@vt.edu
xuchaozhang@microsoft.com

Abstract

Meta-learning has emerged as a trending technique to tackle few-shot text classification and achieve state-of-the-art performance. However, the performance of existing approaches heavily depends on the inter-class variance of the support set. As a result, it can perform well on tasks when the semantics of sampled classes are distinct while failing to differentiate classes with similar semantics. In this paper, we propose a novel **T**ask-**A**daptive **R**eference **T**ransformation (TART) network, aiming to enhance the generalization by transforming the class prototypes to per-class fixed reference points in task-adaptive metric spaces. To further maximize divergence between transformed prototypes in task-adaptive metric spaces, TART introduces a discriminative reference regularization among transformed prototypes. Extensive experiments are conducted on four benchmark datasets and our method demonstrates clear superiority over the state-of-the-art models in all the datasets. In particular, our model surpasses the state-of-the-art method by 7.4% and 5.4% in 1-shot and 5-shot classification on the 20 Newsgroups dataset, respectively. Our code is available at https://github.com/slei109/TART

1 Introduction

Deep learning has achieved great success in many fields but a deficiency of supervised data is often experienced in real-world NLP applications. Few-shot text classification aims to perform classification with a limited number of training instances, which is crucial for many applications but remains to be a challenging task.

Existing approaches for few-shot text classification mainly fall into two categories: i) prompt-based learning (Brown et al., 2020; Gao et al., 2021; Wang et al., 2021), which utilizes Pre-trained Language Models (PLMs) to generate a textual answer in response to a given prompt. Although producing promising results, these methods suffer from (1) requiring a large PLM to function properly; and (2) favoring certain issues which can be naturally posed as a "fill-in-the-blank" problem and do not contain many output classes, rendering them inapplicable in many real-world scenarios. For instance, it is hard to run the large-scale model on devices with limited computing resources, like mobile devices. ii) meta-learning (Finn et al., 2017; Snell et al., 2017), also known as "*learning to learn*": it improves the model's capacity to learn over multiple training tasks, allowing it to quickly adapt to new tasks with only a few training instances. Since meta-learning-based methods (Gao et al., 2019; Bao et al., 2019; Han et al., 2021) rely on learned cross-task transferable knowledge rather than recalling pre-trained knowledge gained through PLMs, these methods have no constraints on the target problem and are broadly studied on the small-scale model, making them more applicable to real-world applications.

Class	Testing Sample	Task 1: Support class: 1,2,3,4		Task 2: Support class: 1,2,3,5	
		MLADA	Ours	MLADA	Ours
1	Animal photos of the week: baby tiger goes for a swim.	1	1	1	1
2	Twitter helps confirm X-shaped bulge at Center of Milky Way.	4	2	2	2
3	Toronto van attack suspect's Facebook post praised misogynist mass killer.	4	3	2	3
4	Apple just solved one of the iphone's most harmful features.	2	4	-	-
5	Apple fritter season is here, and so are the recipes you'll need.	-	-	5	5

Class 1: Environment *Class 2: Science* *Class 3: World News* *Class 4: Tech* *Class 5: Taste*

Figure 1: Prediction results of example tasks with different inter-class variance on the Huffpost dataset. MLADA (Han et al., 2021) performs well on the task with high inter-class variance (e.g., Task 2: *Environment*, *Science*, *World News*, *Taste*), while it fails to distinguish the samples from a task with low inter-class variance (e.g., Task 1: *Environment*, *Science*, *World News*, *Tech*).

Despite the extraordinary effectiveness, we notice that current meta-learning-based approaches may have several limitations. For those methods that learn to represent each class independently in one feature space (Snell et al., 2017; Gao et al.,

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11014–11026
July 9-14, 2023

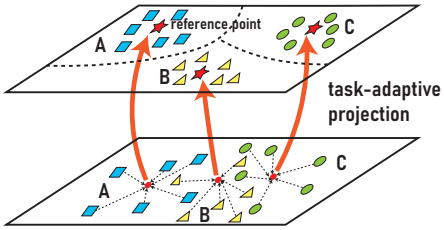


Figure 2: Feature representations in original metric space (bottom) and task-adaptive metric space (top).

2019; Han et al., 2021), their performance is heavily dependent on the inter-class variance of the support set. Specifically, they address the overfitting issue in few-shot learning by directly adopting the hidden features of support samples as a classifier. Thus, they can perform well on tasks when the sampled classes are distinct while failing to differentiate classes with similar semantics. As illustrated in Figure 1, MLADA (Han et al., 2021), which leads to state-of-the-art performance, misclassifies the testing samples of *Science*, *World News* and *Tech* during the testing stage, mostly because *Science* and *Tech* are similar and all three samples contain technology companies, which are difficult to distinguish. If we substitute the support class *Science* with *Taste*, which has clearly different semantic from *Tech*, it can recognize all testing samples except the third one. This example indicates that ignoring task-specific features and treating all tasks identically is inadequate. It is essential to consider the inter-class variance of support sets, particularly when annotated data is scarce. Recently, Bao et al. (2019) leveraged distributional signatures to estimate class-specific word importance. However, it requires extracting relevant statistics of each word from the source pool and the support set for each task, which is time-consuming. A natural question arises: *how can we design an efficient method capable of capturing both cross-task transferable knowledge and task-specific features to enhance the model's generalization ability?*

To tackle these issues, we resort to constructing a task-adaptive metric space via the meta-learning framework. Figure 2 presents the key idea of the proposed method. Intuitively, for comparable classes that cannot be distinguished in the original feature space, if we can project their class prototypes to per-class fixed points, referred to as reference points, in another small space, it is helpful to enhance the divergence between class prototypes in the transformed space. Consequently, we propose a novel Task-Adaptive Reference Transfer Module that uses a linear transformation matrix to project embedding features into a task-specific metric space. In addition, we design a discriminative reference regularization to maximize the distance between transformed prototypes in the task-adaptive metric space for each task. We find the proposed method promotes the model to learn discriminative reference vectors and construct a stable metric space.

Our key contributions can be summarized as follows. 1) We propose a **T**ask-**A**daptive **R**eference **T**ransformation (TART) network for few-shot text classification. The model enhances the generalization by transforming the class prototypes to per-class fixed reference points in task-adaptive metric spaces. 2) We propose a novel discriminative reference regularization to maximize divergence between transformed prototypes in task-adaptive metric spaces to further improve the performance. 3) We evaluate the proposed model on four popular datasets for few-shot text classification. Comprehensive experiments demonstrate that our TART consistently outperforms all the baselines for both 1-shot and 5-shot classification tasks. For instance, our model outperforms MLADA (Han et al., 2021) model by 7.4% and 5.4% in 1-shot and 5-shot classification on the 20 Newsgroups dataset, respectively.

2 Related Work

Few-shot learning Few-shot learning aims to learn a new concept representation from only a few annotated examples. Most existing works can be categorized into three groups: (1) Gradient-based meta-learners, including MAML (Finn et al., 2017), MAML++ (Antoniou et al., 2018), and MetaNets (Munkhdalai and Yu, 2017). The prominent idea is to learn a proper initialization of the neural network, one can expect the network to adapt to novel tasks via backpropagation from limited samples. (2) Graph neural network (Garcia and Bruna, 2017; Liu et al., 2019) based methods, which cast few-shot learning as a supervised message passing task and utilize graph neural networks to train it end-to-end. (3) Metric-based methods (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018), which aim to optimize the transferable embedding using metric learning approaches. Specifically, Matching networks (Vinyals et al., 2016) learns sample-wise metric, where distances

to samples are used to determine the label of the query. Prototypical Networks (Snell et al., 2017) extends the idea from samples to class-wise metric, where all the samples of a specific class are grouped and considered as class prototypes. Then the prototypes are subsequently used for inference.

Transfer learning and Prompt learning for PLMs Few-shot text classification relates closely to transfer learning (Zhuang et al., 2020) that aims to leverage knowledge from source domains to target domains. Fine-tuning Pre-trained Language Models (PLMs) (Devlin et al., 2018; Raffel et al., 2020; Brown et al., 2020; Lei et al., 2022) can also be viewed as a type of transfer learning. Recently, Gao et al. (2021) proposed a prompt-based approach to fine-tune PLMs in a few-shot learning setting for *similar* tasks, which adapts PLMs to producing specific tokens corresponding to each class, instead of learning the prediction head. Meta-learning deviates from these settings by learning to quickly adapt the model to *different* tasks with little training data available (Wang et al., 2021), typically formulated as a N-way K-shot problem.

Few-shot text classification Few-shot text classification has gained increasing attention in recent years. Yu et al. (2018) used an adaptive metric learning approach to select an optimal distance metric for different tasks. Induction Network (Geng et al., 2019) aims to learn an appropriate distance metric to compare validation points with training points and make predictions through matching training points. DMIN (Geng et al., 2020) utilizes dynamic routing to provide more flexibility to memory-based few-shot learning in order to adapt the support sets better. Bao et al. (2019) leveraged distributional signatures (e.g. word frequency and information entropy) to train a model within a meta-learning framework. Another group of methods is to improve performance with the help of additional knowledge, including pre-trained text paraphrasing model (Dopierre et al., 2021; Chen et al., 2022) and class-label semantic information (Luo et al., 2021). Recently, Hong and Jang (2022) constructed a meta-level attention aspects dictionary and determined the top-k most relevant attention aspects to utilize pre-trained models in few-shot learning. MLADA (Han et al., 2021) is an adversarial network, which improves the domain adaptation ability of meta-learning. However, none of these methods consider task-specific features, which is a key factor for few-shot text classification.

3 Model

In this section, we initially discuss the problem setting of few-shot text classification. Then, the overview of the proposed TART is presented in Section 3.2. The technical details for the Task-Adaptive Reference Transfer Module and Discriminative Reference Regularization are described in Sections 3.3 and 3.4, respectively.

3.1 Problem Setting

In N-way K-shot text classification, the objective is to train a model $f_\theta(\cdot)$ that can classify a given query example using the support set $\mathcal{S}$, which comprises K examples for each of the N different classes considered. Note that $f_\theta(\cdot)$ has *not* been pre-trained on any large datasets in advance. In accordance with prior works (Bao et al., 2019; Han et al., 2021), we use the episode training and testing protocols on account of their effectiveness.

Consider that we are given texts from two non-overlapping sets of classes $\mathcal{C}_{train}$ and $\mathcal{C}_{test}$, i.e., $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \emptyset$. The training set $\mathcal{D}_{train}$ is constructed from $\mathcal{C}_{train}$, whereas the test set $\mathcal{D}_{test}$ is derived from $\mathcal{C}_{test}$. The model $f_\theta(\cdot)$ is trained on $\mathcal{D}_{train}$ and evaluated on $\mathcal{D}_{test}$. Both the training set $\mathcal{D}_{train}$ and testing set $\mathcal{D}_{test}$ are comprised of multiple episodes. Each episode consists of a support set $\mathcal{S} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N\times K}$ and a query set $\mathcal{Q} = \{(\boldsymbol{x}_j, y_j)\}_{j=1}^{Q}$, where $\boldsymbol{x}$ represents a text, y is a corresponding class label and Q is the number of query samples. Due to the fact that each episode comprises distinct classes, the model is trained to generalize effectively to few-shot scenarios. After meta-training is completed, we evaluate the performance of its few-shot text classification on the test set $\mathcal{D}_{test}$ over all the episodes. For better understanding, we denote "episode" as "task" in the following context.

3.2 Overview

In this work, we resort to constructing a task-adaptive metric space to boost the performance of few-shot text classification. In contrast to previous approaches that construct the metric space using task-agnostic features, we propose to construct a task-adaptive metric space that enlarges the relative differences among sampled classes within a task. Figure 3 illustrates an overview of the proposed TART model. Using a shared feature extractor, we encode contextual embeddings of the support and query texts for each episode. Each

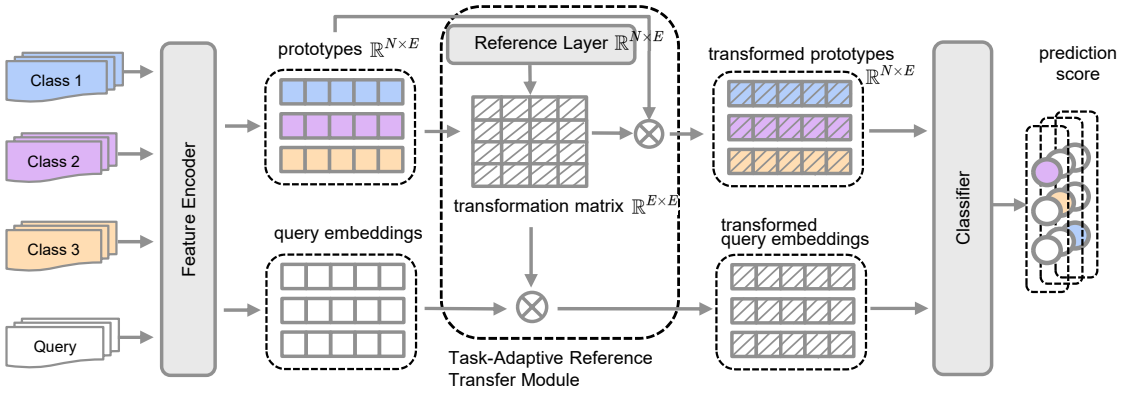


Figure 3: Illustration of the pipeline of TART for a 3-way 3-shot task with three query examples. After obtaining the embeddings of support and query inputs, the Task-Adaptive Reference Transfer Module is introduced to transform the embedding features into task-specific ones with a linear transformation matrix. Then, the query texts are classified by measuring the distance between each transformed prototype and transformed query embeddings in the task-adaptive metric space.

class prototype is produced by averaging the support contextual embeddings. Then, we offer a novel module, dubbed Task-Adaptive Reference Transfer Module, to construct a task-adaptive metric space and project contextual embeddings from the task-agnostic space to task-specific ones. The classification of the query texts is accomplished by assigning each text the category of the closest prototype in the newly generated task-specific metric space. To learn discriminative reference vectors and construct a stable metric space, we also propose Discriminative Reference Regularization (DRR), which measures the distance between transformed prototypes in the task-adaptive metric space.

3.3 Task-Adaptive Reference Transfer Module

The key idea of the Task-Adaptive Reference Transfer Module is to acquire a feature transformer to construct a task-adaptive metric space. Intuitively, for comparable classes that cannot be distinguished in the original feature space, if we can project their class prototypes to per-class fixed points, referred to as reference points, in another small space, it is helpful to enhance the divergence between class prototypes in the transformed space. Below, we describe how to construct a task-adaptive metric space and make a classification based on it.

Different from learning a non-linear transformation matrix directly, our model adopts a linear transformation matrix calculated by using the reference layer and the prototypes of the support set. This can effectively avoid overfitting since it introduces fewer learnable parameters. First, we introduce a set of reference vectors $\{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N\}$ as the fixed points for the transformed space, which are learned via a linear layer, dubbed reference layer. We use the weight matrix of the reference layer and the prototype set of the support contextual embedding to compute the transformation matrix. Formally, let R represent the weight matrix of the reference layer and P denote the prototype matrix of the support texts. We construct the transformation matrix W by finding a matrix such that $PW = R$.

Specifically, given a N-way K-shot episode (task), each class prototype is obtained by averaging the support contextual embeddings:

$$\boldsymbol{p}_c = \frac{1}{|\mathcal{S}_c|} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{S}_c} f_\theta(\boldsymbol{x}_i), \tag{1}$$

where S_c denotes the support samples for the class c. Accordingly, the reference weight matrix R is defined as $[\frac{\boldsymbol{r}_1}{\|\boldsymbol{r}_1\|}, \ldots, \frac{\boldsymbol{r}_N}{\|\boldsymbol{r}_N\|}]$, where $R \in \mathbb{R}^{N\times E}$. Note that each row in R is the per-class reference vector and is learned during the training stage. In general, P is a non-square matrix and we can calculate its generalized inverse (Ben-Israel and Greville, 2003) with $P^+ = \{P^T P\}^{-1} P^T$. Thus, the transformation matrix is computed as $W = P^+ R$, where $W \in \mathbb{R}^{E\times E}$.

For each query input, we calculate the probability of belonging to class c by applying a softmax function over the distance between the transformed query embeddings and each transformed prototype in the task-adaptive metric space. Concretely, given a distance function d, for each query input $\boldsymbol{x}_q$ and

prototype set $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N\}$, we have

$$p(y = c|\boldsymbol{x}_q) = \frac{\exp\left(-d(f_\theta(\boldsymbol{x}_q)W, \boldsymbol{p}_c W)\right)}{\sum_{\boldsymbol{p}_c \in \mathcal{P}} \exp\left(-d(f_\theta(\boldsymbol{x}_q)W, \boldsymbol{p}_c W)\right)} \tag{2}$$

The distance function d commonly adopts the cosine distance or squared Euclidean distance. Learning proceeds by minimizing the classification loss $\mathcal{L}_{cls}$, which is formatted as:

$$\mathcal{L}_{cls} = \frac{1}{|\mathcal{Q}|} \sum_{\boldsymbol{x}_q \in \mathcal{Q}} [d(f_\theta(\boldsymbol{x}_q)W, \boldsymbol{p}_c W) + \log \sum_{\boldsymbol{p}_c \in \mathcal{P}} \exp\left(-d(f_\theta(\boldsymbol{x}_q)W, \boldsymbol{p}_c W)\right)] \tag{3}$$

3.4 Discriminative Reference Regularization

To further improve TART, we propose a Discriminative Reference Regularization (DRR) for more discriminative metric spaces. Since the transformation matrix is only decided by the reference layer and the prototype set of the given task, these task-independent reference vectors are the key elements to constructing discriminative metric spaces. For training the reference vectors, we propose to maximize the distance between all transformed prototypes in the task-adaptive metric spaces during training. Different from contrastive learning, our DRR requires no additional data and focuses more on learning task-independent reference vectors instead of the feature encoder for the downstream task. Formally, for a particular episode, given the prototype set $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N\}$ and the transformation matrix W, the discriminative loss $\mathcal{L}_{drr}$ is defined as:

$$\mathcal{L}_{drr} = \sum_{i \neq j, \boldsymbol{p} \in \mathcal{P}} -d(\boldsymbol{p}_i W, \boldsymbol{p}_j W) \tag{4}$$

The total loss for training our TART model is thus $\mathcal{L} = \mathcal{L}_{cls} + \lambda \mathcal{L}_{drr}$, where λ serves as regularization strength. Empirically, we set $\lambda = 0.5$ in our experiments.

For better understanding, the whole training procedure for TART is summarized in Algorithm 1. The model parameters and reference layers are randomly initialized. Given each training episode, we randomly chose T episodes of the support set and query set from the training dataset, each episode consists of K labeled samples over N classes. Then, with the support set $\mathcal{S}_c$ for class c, the prototype $\boldsymbol{p}_c$ is obtained for each class (in line 5). Based on the prototype set and the reference layers, the transformation matrix W is computed as a task-adaptive projection matrix (in lines 7-9). For each query input, the distances between the transformed query embeddings and each transformed prototype are measured in the task-adaptive metric space, and the classification loss $\mathcal{L}_{cls}$ is computed using these distances (in lines 10-12). The discriminative loss is obtained over the prototype set for each episode (in line 15). The learnable parameters of the feature encoder and the reference layers are updated based on the total loss $\mathcal{L}$ (in line 16). This process gets repeated for every remaining episode with new classes of texts and queries.

Algorithm 1 TART Training Procedures

Input: A feature encoder f_θ, a training set $\mathcal{D}_{train} = \{(\mathcal{S}_1, \mathcal{Q}_1), \ldots, (\mathcal{S}_T, \mathcal{Q}_T)\}$, reference layers $\{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N\}$.

1: Randomly initialize the model parameters and reference layers.
2: **for** each episode $(\mathcal{S}_i, \mathcal{Q}_i) \in \mathcal{D}_{train}$ **do**
3: $\quad \mathcal{L}_{cls} \leftarrow 0, \mathcal{L}_{drr} \leftarrow 0$
4: $\quad$ **for** k in $\{1, \ldots, N\}$ **do**
5: $\quad\quad \boldsymbol{p}_c \leftarrow \frac{1}{|\mathcal{S}_c|} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{S}_c} f_\theta(\boldsymbol{x}_i)$
6: $\quad$ **end for**
7: $\quad R \leftarrow [\frac{\boldsymbol{r}_1}{\|\boldsymbol{r}_1\|}, \ldots, \frac{\boldsymbol{r}_N}{\|\boldsymbol{r}_N\|}]$
8: $\quad P_i \leftarrow [\frac{\boldsymbol{p}_1}{\|\boldsymbol{p}_1\|}, \ldots, \frac{\boldsymbol{p}_N}{\|\boldsymbol{p}_N\|}]$
9: $\quad W_i = \{P_i^T P_i\}^{-1} P_i^T R$
10: $\quad$ **for** k in $\{1, \ldots, N\}$ **do**
11: $\quad\quad$ **for** $(\boldsymbol{x}, y)$ in $\mathcal{Q}_i$ **do**
12: $\quad\quad\quad$ Compute $\mathcal{L}_{cls}$ using Eq.3
13: $\quad\quad$ **end for**
14: $\quad$ **end for**
15: $\quad$ Compute $\mathcal{L}_{drr}$ using Eq.4
16: $\quad$ Update model parameters minimizing $\mathcal{L}$ via optimizer
17: **end for**

4 Experiments

4.1 Datasets

We use four benchmark datasets for the evaluation of few-shot text classification task, whose statistics are summarized in Table 1.

HuffPost headlines consists of news headlines published on HuffPost between 2012 and 2018 (Misra, 2018). These headlines are split into 41 classes. In addition, their sentences are shorter and less grammatically correct than formal phrases.

Amazon product data contains product reviews from 24 product categories, including 142.8 million reviews spanning 1996-2014 (He and McAuley, 2016). Our task is to identify the product categories of the reviews. Due to the huge size of the original dataset, we sample a subset of 1,000 reviews from each category.

Reuters-21578 is collected from Reuters arti-

Dataset	# samples	Avg. # tokens/sample	Vocab size	# train/val/test classes
Huffpost	36,900	11	8,218	20/5/16
Amazon	24,000	140	17,062	10/5/9
Reuters	620	168	2,234	15/5/11
20 Newsgroups	18,820	340	32,137	8/5/7

Table 1: Statistics of the four benchmark datasets. *Avg. # tokens/sample* denotes the average tokens per sample.

cles in 1987 (Lewis, 1997). We use the standard ApteMode version of the dataset. Following Bao et al. (2019), we evaluate 31 classes and eliminate articles with multiple labels. Each class comprises a minimum of twenty articles.

20 Newsgroups is a collection of approximately 20,000 newsgroup documents (Lang, 1995), partitioned equally among 20 different newsgroups.

4.2 Baselines.

We compare our TART with multiple competitive baselines, which are briefly summarized as follows: (i) **MAML** (Finn et al., 2017) is trained by maximizing the sensitivity of the loss functions of new tasks so that it can rapidly adapt to new tasks once the parameters have been modified via a few gradient steps. (ii) **Prototypical Networks** (Snell et al., 2017), abbreviated as PROTO, is a metric-based method for few-shot classification by using sample averages as class prototypes. (iii) **Latent Embedding Optimization** (Rusu et al., 2018), abbreviated as LEO, learns a low-dimensional latent embedding of model parameters and performs gradient-based meta-learning in this space. (iv) **Induction Networks** (Geng et al., 2019) learns a class–wise representation by leveraging the dynamic routing algorithm in meta-learning. (v) **HATT** (Gao et al., 2019) extends PROTO by adding a hybrid attention mechanism to the prototypical network. (vi) **DS-FSL** (Bao et al., 2019) maps the distribution signatures into attention scores to extract more transferable features. (vii) **MLADA** (Han et al., 2021) adopts adversarial networks to improve the domain adaptation ability of meta-learning. (viii) **Frog-GNN** (Xu and Xiang, 2021) extracts better query representations with multi-perspective aggregation of graph node neighbors. (ix) **P-Tuning** (Liu et al., 2021) is a prompt-based method that employs soft-prompting techniques to optimize prompts in continuous space. (x) **LEA** (Hong and Jang, 2022) determines the top-k most relevant attention aspects to utilize pre-trained models in few-shot learning.

4.3 Implementation Details

In accordance with prior work (Bao et al., 2019), we use pre-trained fastText (Joulin et al., 2016) for word embedding. As a feature extractor, we employ a BiLSTM with 128 hidden units and set the number of hidden units for the reference layers to 256. We take cosine similarity as the distance function. The model is implemented in PyTorch (Paszke et al., 2017) using the Adam (Kingma and Ba, 2014) optimizer with a 10^{-4} learning rate. For the sake of a fair comparison, we follow the identical evaluation protocol and train/val/test split as Bao et al. (2019). The model parameters and reference layers are randomly initialized. During meta-training, we perform 100 training episodes per epoch. Meanwhile, we apply early stopping if the accuracy on the validation set does not increase after 20 epochs. We evaluate the model performance based on a total of one thousand testing episodes and present the average accuracy across five different random seeds. All experiments are conducted with NVIDIA V100 GPUs.

4.4 Comparisons

The experimental results are shown in Table 2 in terms of various datasets, methods, and few-shot settings. As demonstrated in Table 2, our model outperforms recent methods across all datasets, with the exception of Amazon's 1-shot setting. In particular, our model achieves an average accuracy of 69.0% for 1-shot classification and 82.3% for 5-shot classification. Our model surpasses the state-of-the-art approach MLADA (Han et al., 2021) by an average of 5.1% in 1-shot and 0.9% in 5-shot, demonstrating the effectiveness of task-adaptive metric space. Specifically, our method delivers a substantial improvement, 9.9% in 1-shot on Reuters, and 7.4% and 5.4% in 1-shot and 5-shot on 20Newsgroup, respectively. The average length of texts in these datasets is longer than in the other datasets, verifying its superiority in the longer texts. Moreover, we show that our model achieves a more significant boost in the 1-shot than in the 5-shot, indicating that our model contributes more

Method	HuffPost		Amazon		Reuters		20 News		Average	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
MAML (2017)	35.9	49.3	39.6	47.1	54.6	62.9	33.8	43.7	40.9	50.8
PROTO (2017)	35.7	41.3	37.6	52.1	59.6	66.9	37.8	45.3	42.7	51.4
LEO* (2018)	28.8	42.3	39.5	52.5	35.4	54.1	36.4	52.2	35.0	50.3
Induct (2019)	38.7	49.1	34.9	41.3	59.4	67.9	28.7	33.3	40.4	47.9
HATT (2019)	41.1	56.3	49.1	66.0	43.2	56.2	44.2	55.0	44.4	58.4
DS-FSL (2020)	43.0	63.5	62.6	81.1	81.8	96.0	52.1	68.3	59.9	77.2
MLADA (2021)	45.0	64.9	68.4	**86.0**	82.3	96.7	59.6	77.8	63.9	81.4
LEA (2022)	46.2	65.8	66.5	83.5	69.0	89.0	54.1	60.2	58.9	74.6
TART w/o DRR	**48.4**	66.0	68.9	83.5	90.4	96.2	66.4	82.2	68.5	81.9
TART	46.9	**66.8**	**70.1**	82.4	**92.2**	**96.7**	**67.0**	**83.2**	**69.0**	**82.3**

Table 2: Results of 5-way 1-shot and 5-way 5-shot classification on four datasets. The bottom two rows present our ablation study. *Reported by Hong and Jang (2022).

Method	PLM	EK	HuffPost		Amazon		Reuters		20 News		Average	
			1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LEA	×	×	48.4	**71.6**	63.6	82.7	71.6	83.1	53.5	65.9	59.3	75.8
Frog-GNN	×	×	54.1	69.6	71.5	83.6	-	-	-	-	-	-
P-Tuning	✓	×	**54.5**	65.8	62.2	79.1	**90.0**	**96.7**	56.2	77.7	65.7	79.8
ContrastNet	×	✓	53.1	65.3	**76.1**	**85.2**	86.4	95.3	71.7	81.6	**71.8**	81.9
TART	×	×	46.5	68.9	73.7	84.3	86.9	95.6	**73.2**	**84.9**	70.1	**83.4**

Table 3: 5-way 1-shot and 5-way 5-shot classification on four datasets using BERT. *PLM* denotes prompting language model and *EK* denotes extra knowledge. Note that *ContrastNet* utilizes a pre-trained short-texts paraphrasing model to generate data augmentation of texts.

to a generation of distinguishable class representation, particularly when the labeled class sample is limited.

4.5 Ablation Study

We conduct extensive studies to examine the effects of DRR, contextualized representations and reference vectors.

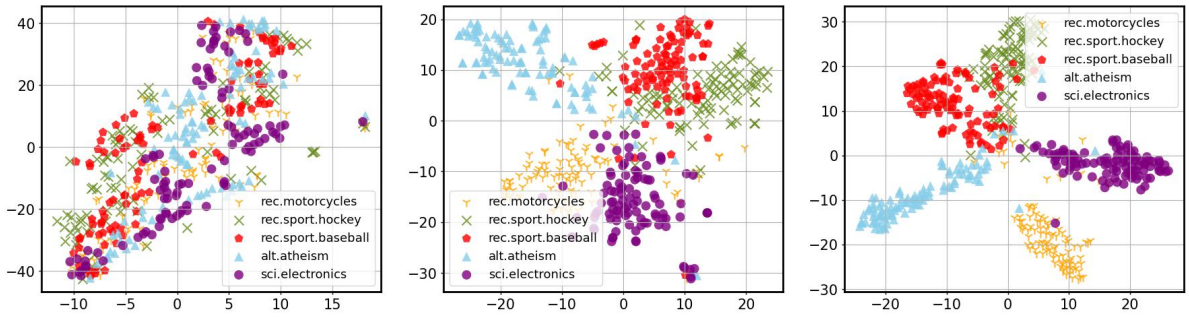
First, we study how the DRR affects the performance of our model. The results are presented at the bottom of Table 2. With the use of DRR, the model can construct a more discriminative subspace for classification, especially in 1-shot settings. This empirical study validates the effectiveness of DRR in enhancing performance.

We also experiment with contextualized representations, given by the pure pre-trained `bert-base-uncased` model, dubbed BERT$_{BASE}$ (Devlin et al., 2018). The results are shown in Table 3. We observe that BERT improves classification performance for the text-level dataset. Even while ContrasNet requires a pre-trained short-texts paraphrasing model to generate data augmentation, our model can outperform it without requiring any additional knowledge on the 5-shot setting.

The introduction of the reference vectors is to enhance the divergence between class prototypes in the metric space. Even though adding more layers to the feature encoder could theoretically make it better, the small number of labeled samples is probably causing it to overfit. Moreover, we investigate the performance of the feature encoder with multiple layers. We adopt MLADA as the basic model, which leads to state-of-the-art performance. The results are shown in Table 4. We found that the feature encoder with two layers of Bi-LSTM achieves better performance than one layer in a 1-shot setting. But the accuracy decreases when the number of layers increases further. In contrast, our model uses a linear transformation matrix that is figured out by using the reference layer and the prototypes of the support set. This can effectively enhance generalization while avoiding overfitting since it introduces fewer learnable parameters.

4.6 Hyperparameter Analysis

We analyze the effect of different settings of hyperparameter λ. Table 5 demonstrates the accuracy in different settings on the validation set of the Reuters and 20 Newsgroup datasets. We discover that $\lambda = 0.5$ yields the optimum performance, and



(a) AVG (b) MLADA (c) TART

Figure 4: t-SNE visualization of the input representation of the classifier for a testing episode (N = 5, K = 5, Q = 100) sampled from 20 Newsgroups. Note that the 5 classes are not seen in training set. The input representation of the classifier is given by (a) the average of word embeddings (b) MLADA and (c) TART (ours).

Method	HuffPost		Amazon		Reuters		20 News		Average	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
1 layer Bi-LSTM	45.0	64.9	68.4	**86.0**	82.3	96.7	59.6	77.8	63.9	81.4
2 layer Bi-LSTM	45.2	65.2	67.1	83.7	85.5	96.4	64.0	78.6	65.5	81.0
3 layer Bi-LSTM	45.4	63.6	66.0	83.2	84.3	**97.9**	64.4	78.5	65.0	80.8
TART	**46.9**	**66.8**	**70.1**	82.4	**92.2**	96.7	**67.0**	**83.2**	**69.0**	**82.3**

Table 4: Comparison of the feature encoder with different numbers of layers.

Settings	Reuters		20 News	
	1-shot	5-shot	1-shot	5-shot
$\lambda = 0.3$	91.5	96.3	66.6	82.9
$\lambda = 0.5$	**92.2**	**96.7**	**67.0**	**83.2**
$\lambda = 0.7$	89.5	95.4	66.1	82.0
$\lambda = 0.9$	89.1	94.9	65.7	81.7

Table 5: Evaluation accuracy on the validation set of Reuters and 20 Newsgroup datasets. Different settings adjust the proportion of $\mathcal{L}_{drr}$.

further reduction/increase in the ratio lead to performance degradation. It is likely because $\mathcal{L}_{drr}$ can improve the divergence of the class prototypes. But a too-large ratio of $\mathcal{L}_{drr}$ would make the model focus more on the task-independent reference vectors while ignoring the learning for a unique feature space, which may lead to an over-fitting problem.

4.7 Visualization

We utilize visualization experiments to demonstrate that our model can build high-quality sentence embeddings and identify significant lexical features for unseen classes.

To illustrate that our model can generate high-quality sentence embeddings for unseen classes, we view the high-dimensional features as two-dimensional images using the t-SNE algorithm (Van der Maaten and Hinton, 2008). Figure 4 depicts the 256-dimension feature representations for a 5-way 5-shot testing episode sampled from the 20 NewsGroup dataset. From the results, it is evident that the distances between the inter-classes are much larger than those of the average word embeddings and MLADA depicted in Figure 4(a) and Figure 4(b), respectively. This enlarged inter-class spacing shows that our method can construct a more distinct feature space for each episode.

In addition, the weight vectors on the same support samples are depicted in two testing episodes. The example is drawn from the Huffpost dataset. Figure 5 demonstrates that our apporach is capable of generating task-specific attention. Even with the same text, the attention of each word varied based on the different combinations of categories in the task. Specifically, as compared to *Science* and *Education* class, Word “Crisis", “attack" and “climb" become more important for *World News*, *Tech* and *Education* class, respectively.

5 Conclusion

In this work, we propose a novel TART for few-shot text classification, which can enhance the generalization by transforming the class prototypes to per-class fixed reference points in task-adaptive

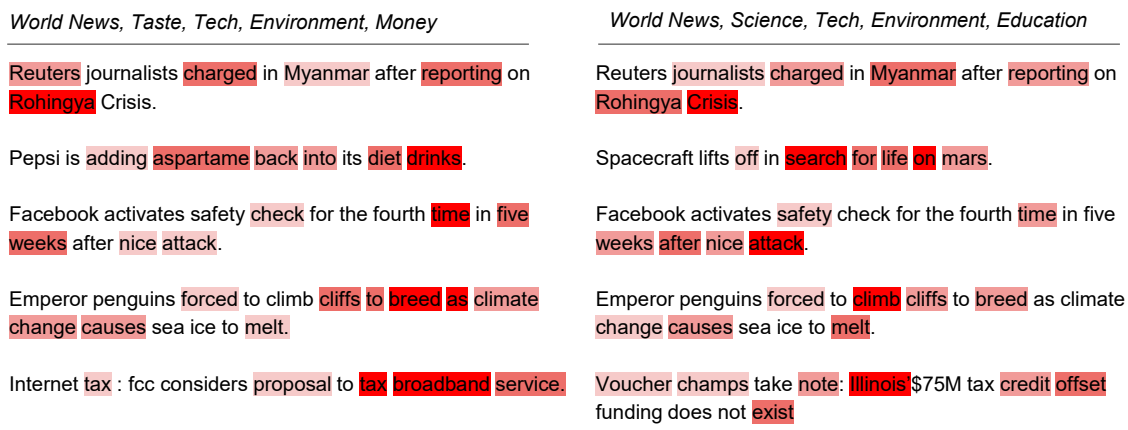


Figure 5: The visualization of task-specific attention weights generated by our model. We visualize our model's support sets of two different tasks (5-way 1-shot) in Huffpost dataset. Word "Crisis" is downweighed for *World News* class when compared to *Taste*, *Tech*, *Environment* and *Money* classes (left), but it becomes important when replacing *Taste* and *Money* with *Science* and *Education* (right).

metric spaces. Specifically, a task-adaptive transfer module is designed to project embedding features into a task-specific metric space by using a linear transformation matrix. In addition, we propose a discriminative reference regularization to maximize divergence between transformed prototypes in task-adaptive metric spaces. The proposed model is evaluated on four standard text classification datasets. Without any extra knowledge or data information, our TART outperforms previous work by a large margin.

6 Limitations

Our approach is based on meta-learning and is designed for constrained situations where computing resources are limited, such as on-device settings. Therefore, using large and complex feature encoders like LLM may pose scalability challenges. In addition, if the task involves a significant number of new classes, the model may not scale effectively. Lastly, our method is primarily suitable for text classification, such as news category or product review classification. It is not appropriate for text generation tasks.

References

Antreas Antoniou, Harrison Edwards, and Amos Storkey. 2018. How to train your maml. In *International Conference on Learning Representations*.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2019. Few-shot text classification with distributional signatures. *arXiv preprint arXiv:1908.06039*.

Adi Ben-Israel and Thomas NE Greville. 2003. *Generalized inverses: theory and applications*, volume 15. Springer Science & Business Media.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xu. 2022. Contrastnet: A contrastive learning framework for few-shot text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10492–10500.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Thomas Dopierre, Christophe Gravier, and Wilfried Logerais. 2021. Protaugment: Unsupervised diverse short-texts paraphrasing for intent detection meta-learning. *arXiv preprint arXiv:2105.12995*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks

for noisy few-shot relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6407–6414.

Victor Garcia and Joan Bruna. 2017. Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043*.

Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Dynamic memory induction networks for few-shot text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1087–1094.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. *arXiv preprint arXiv:1902.10482*.

Chengcheng Han, Zeqiu Fan, Dongxiang Zhang, Minghui Qiu, Ming Gao, and Aoying Zhou. 2021. Meta-learning adversarial domain adaptation network for few-shot text classification. *arXiv preprint arXiv:2107.12262*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

SK Hong and Tae Young Jang. 2022. Lea: Meta knowledge-driven self-attentive document embedding for few-shot text classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 99–106.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier.

Shuo Lei, Xuchao Zhang, Jianfeng He, Fanglan Chen, and Chang-Tien Lu. 2022. Uncertainty-aware cross-lingual transfer with pseudo partial labels. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1987–1997, Seattle, United States. Association for Computational Linguistics.

David Lewis. 1997. Reuters-21578 text categorization test collection, distribution 1.0. *http://www. research/. att. com*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Y Liu, J Lee, M Park, S Kim, E Yang, SJ Hwang, and Y Yang. 2019. Learning to propagate labels: Transductive propagation network for few-shot learning. In *7th International Conference on Learning Representations, ICLR 2019*.

Qiaoyang Luo, Lingqiao Liu, Yuhao Lin, and Wei Zhang. 2021. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2773–2782.

Rishabh Misra. 2018. News category dataset. *DOI: DOI: https://doi. org/10.13140/RG*, 2(20331.18729).

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *International Conference on Machine Learning*, pages 2554–2563. PMLR.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2018. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Chengyu Wang, Jianing Wang, Minghui Qiu, Jun Huang, and Ming Gao. 2021. Transprompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2792–2802.

Shiyao Xu and Yang Xiang. 2021. Frog-gnn: Multi-perspective aggregation based graph neural network for few-shot text classification. *Expert Systems with Applications*, 176:114795.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. *arXiv preprint arXiv:1805.07513*.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

ACL 2023 Responsible NLP Checklist

A For every submission:

☐ A1. Did you describe the limitations of your work?
Left blank.

☐ A2. Did you discuss any potential risks of your work?
Left blank.

☐ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☐ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☐ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☐ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☐ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Left blank.