Metric Sentiment Learning for Label Representation

Chengyu Song Fei Cai* Jianming Zheng Wanyu Chen Zhiqiang Pan
Science and Technology on Information Systems Engineering Laboratory,
National University of Defense Technology, Changsha, China
{songchengyu, caifei08, zhengjianming12, wanyuchen, panzhiqiang}@nudt.edu.cn

ABSTRACT

Label representation aims to generate a so-called verbalizer to an input text, which has a broad application in the field of text classification, event detection, question answering, etc. Previous works on label representation, especially in a few-shot setting, mainly define the verbalizers manually, which is accurate but time-consuming. Other models fail to correctly produce antonymous verbalizers for two semantically opposite classes. Thus, in this paper, we propose a metric sentiment learning framework (MSeLF) to generate the verbalizers automatically, which can capture the sentiment differences between the verbalizers accurately. In detail, MSeLF consists of two major components, i.e., the contrastive mapping learning (CML) module and the equal-gradient verbalizer acquisition (EVA) module. CML learns a transformation matrix to project the initial word embeddings to the antonym-aware embeddings by enlarging the distance between the antonyms. After that, in the antonym-aware embedding space, EVA first takes a pair of antonymous words as verbalizers for two opposite classes and then applies a sentiment transition vector to generate verbalizers for intermediate classes. We use the generated verbalizers for the downstream text classification task in a few-shot setting on two publicly available fine-grained datasets. The results indicate that our proposal outperforms the state-of-the-art baselines in terms of accuracy. In addition, we find CML can be used as a flexible plug-in component in other verbalizer acquisition approaches.

CCS CONCEPTS

• **Information systems** → *Sentiment analysis.*

KEYWORDS

label representation; pre-training; metric learning

ACM Reference Format:
Chengyu Song, Fei Cai, Jianming Zheng, Wanyu Chen and Zhiqiang Pan. 2021. Metric Sentiment Learning for Label Representation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482369

*Corresponding author.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482369

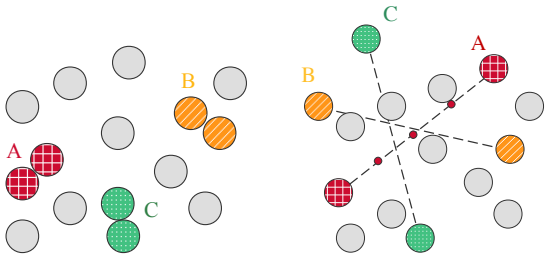
1 INTRODUCTION

Label representation targets to verbalize a natural text to make the machine understand the semantics, which has been widely used in the state-of-the-art transfer learning models to transform various natural language processing tasks into the text inference tasks [35, 36]. It has led to significant improvements, especially in the few-shot scenarios [7]. In such settings, the labels are represented as the model-selected words, i.e., verbalizers [43], and the pre-trained language model is fine-tuned to maximize the probability of selecting the accurate verbalizers. Many existing NLP methods apply verbalizers to aid the downstream tasks, e.g., event extraction [10], text classification [43, 52] and question answering [13, 54], where generating the verbalizer mainly depends on the input for a specific task. For instance, for a case in the Yelp-Review dataset, a classifier is trained to generate one of the words {"Great", "Good", "Okey", "Bad" and "Terrible"}, corresponding to the labels {1, 2, 3, 4, 5}, which is taken as a text generation task. It has been experimentally demonstrated that the verbalizers have a significant impact on the downstream tasks, especially in the few-shot scenarios [7].

Previous approaches for representing the labels, i.e., generating the verbalizers, mainly rely on a fixed pattern to match the words with the labels. They simply refer to the common sense to determine a static list of highly contrastive words (i.e., antonyms) that correspond to different labels one by one [10, 12, 19, 29]. Such approaches make sense but hardly adapt to the dynamic situation of natural language, failing to generate appropriate verbalizers corresponding to various inputs. In addition, a matching based approach is proposed to generate the verbalizer of an individual class by calculating the probability of the words matching a particular class of data [42, 43].

However, the verbalizers generated by such approaches are often unsatisfied. First, they cannot ensure the generated verbalizers for two opposite classes are antonyms. Second, the generated verbalizers fail to accurately express the differences among the intermediate classes apart from these two opposite classes. It could be due to ignoring the overlap of embedding spaces caused by the distributional assumption of word co-occurrence [50, 53, 55], i.e., words in similar contexts usually have similar embeddings. We visualize the embeddings of three pairs of antonymous words in Fig. 1, which are picked up from RoBERTa-Large [21]. The red (A), yellow (B) and green (C) circles represent the antonym pairs in the embedding space. It can be seen from Fig. 1a that the words keep close to their corresponding antonyms, i.e., creating an overlap in the embedding space. However, we argue this overlap will lead to difficulty to generate correct verbalizers. Instead, the antonymous words should be highly contrastive as plotted in Fig. 1b.

Hence, in this paper, we would like to solve the aforementioned issues by borrowing the merits of metric learning [3, 45, 48, 49]



(a) Problem of the overlap of embedding space.

(b) Antonym-awere embedding distribution.

Figure 1: Visualization of words from the pre-trained language model RoBERTa-Large.

as well as the manifold learning [5, 9, 40] to automatically generate the verbalizers for an input text. We propose a **Metric Sentiment Learning Framework (MSeLF)** that consists of two major components, i.e., the Contrast Map Learning (CML) module and the Equal-gradient Verbalizer Acquisition (EVA) module. CML performs like a dimensional transformation operation on the original word embeddings, which is trained with a triplet loss to create an antonym-aware embedding space that well solves the antonym overlap problem in the embedding space. Then, EVA applies a modified matching method to select a pair of antonyms as the verbalizers of two opposite classes, and refers to a sentiment transition vector in the antonym-aware embedding space for generating the verbalizers for the intermediate classes.

To sum up, our contributions can be summarized as follows:

(1) To the best of our knowledge, we are the first to use the metric learning and the manifold learning for automatically generating the verbalizers by proposing the metric sentiment learning framework (MSeLF).
(2) We use CML to create an antonym-aware embedding space and apply the sentiment transition vectors to generate the accurate representation of the label for each class, which solves the overlap problem in the embedding space.
(3) We apply the verbalizers generated by our proposal as well as the state-of-the-art baselines in a few-shot setting for the text classification task on two public fine-grained datasets, i.e., Yelp-Review [1] and Amazon-Review [2]. The results show that MSeLF outperforms the baselines and presents a close performance to the manual evaluation.

2 RELATED WORK

We first review the traditional approaches for label representation in Sec. 2.1, which focus on representing labels as the model-selected words. Then, we summarize recent works on antonym distinction in Sec. 2.2, which is widely applied to few-shot learning models for generating high-quality word embeddings. Finally, we summarize the related works on metric learning in Sec. 2.3, which is often used to avoid the overfitting problem in few-shot learning models.

2.1 Label representation approaches

In recent state-of-the-art transfer learning methods, labels are represented as words to be generated for the text classification task, which is viewed as a text generation task. Existing methods for label-word matching can be categorized to the pattern-based methods and the matching-based models.

On the one hand, the pattern-based methods focus on replacing labels with fixed words following a pre-defined pattern. For example, Du and Cardie [10], Liu et al. [19] propose to target an event extraction to a question word, e.g., "where" and "when", corresponding to the extraction type of location and time, respectively. Moreover, Hu et al. [12] utilize certain words to represent entities that may be contained in the image and use the correlations between words to aid the multi-label classification of images. Furthermore, Nogueira et al. [29] investigate the effect of label representation on document retrieval and ranking, where "Positive" and "Negative" are replaced with "True" and "False", respectively.

On the other hand, the matching-based methods rely on the scoring criteria to identify the representative words for different categories. For instance, Petroni et al. [31], Schick and Schütze [43] propose to assign a random word to each label for initializing the label-word mapping, which is then improved iteratively by finding a better inversion of word substitution via a greedy fashion. Moreover, Schick et al. [42] utilize one Vs. Rest to convert multi-category matches to two-category matches and adopt the likelihood ratios to measure how well the verbalizer matches the instance. However, the above approaches either fail to well adapt the dynamic situation of natural language or cannot ensure the generation of words with sufficient contrast for classes with opposite contexts. Instead, our proposal is able to produce appropriate verbalizers for different inputs. Moreover, by taking the antonyms as verbalizers, MSeLF can fully exploit the antonym-aware embedding space to make verbalizers close to the manual setting.

2.2 Antonym distinction approaches

In general, extracting verbalizer is sensitive to the representation of antonym pairs, where the overlapped word embeddings lead difficulty to distinguish the antonym pairs. Existing approaches on antonym distinction can be divided into two types [2], i.e., the embedding-based and the pattern-based approaches.

The embedding-based approaches heavily rely on the distributional assumption that words with similar or opposite meanings usually occur in the respectively similar or opposite contexts [47]. For instance, Cho et al. [8], Mikolov et al. [23, 24] use the neural networks to train the model. In addition, the large-scale texts are often used to train the embedding vectors [4, 32, 39]. For example, Scheible et al. [41] explain the difference in the distribution of synonyms and antonyms by constructing a distributional word space model. Adel and Schütze [1] adopt the co-reference chains to generate the text embeddings with antonym classification capabilities. Moreover, Nguyen et al. [27] distinguish the antonyms from synonyms by integrating the distributional lexical comparison information. The pattern-based approaches focus on obtaining lexical and syntactic patterns from a large-scale text corpora [18, 26, 44]. For example, Roth and im Walde [38] utilize the discourse relations as additional features of lexical syntactic patterns to distinguish the antonyms from synonyms. Nguyen et al. [28] combine the lexicon-syntactic pattern features with the distributional features to represent the antonyms.

[1] https://www.kaggle.com/yelp-dataset/yelp-dataset
[2] https://www.kaggle.com/snap/amazon-fine-food-reviews

However, the aforementioned models mainly focus on using either the synonyms or antonyms for training. Our proposal refers to a triplet network to ensure that both synonyms and antonyms are involved in training. In addition, upon a pre-trained language model, i.e., RoBERTa-Large [21], our approach can well exploit the word semantics.

2.3 Metric learning methods

Metric learning has been widely applied to few-shot learning to derive transferable knowledge and works well on the target task without fine-tuning [49]. Metric-based methods generally learn a distance function between the data points to classify the instances to the labeled samples. The distance function consists of two major parts, i.e., an embedding function encoding any instance into a representation space and a similarity metric for classifying instances into categories. For example, Koch et al. [14], Loster et al. [22] use a Siamese network which takes two instances as inputs and outputs a scalar indicating whether the inputs belong to the same class or not [30, 34]. Vinyals et al. [48] propose the first metric-based meta-learning algorithm, which is essentially a parametric nearest neighbors algorithm [6, 15, 51]. Moreover, Snell et al. [45] use the class representations rather than the example representations in the support set and utilize a similarity metric to classify the instances [20]. Sung et al. [46] propose the relation network, which consists of an embedding function and a scoring function. Unlike the prototype network, the scoring function is defined as a deep neural network rather than a measurement of the Euclidean distance.

However, existing metric learning methods merely focus on measuring the distance between the instances and the class representations, without fully considering the transition relationship between the instances. Instead, our proposal adopts a sentiment transition vector to measure the relative sentiment shifts between words so as to obtain the accurate label representations.

3 APPROACH

In this section, we first define the task investigated in this paper and introduce the framework of our proposal in Sec. 3.1. Then, we detail the contrast map learning (CML) module in Sec. 3.2 and the equal-gradient verbalizer acquisition (EVA) part in Sec. 3.3.

3.1 Task definition and model framework

Before defining the verbalizer acquisition task in detail, we first introduce a situation where the verbalizer works, i.e., a text reformulation process proposed by a semi-supervised few-shot learning model called PET [43], which reformulates the sentences with manually designed patterns. Formally, given a text training dataset $\mathcal{T} = \{t_i\}_{i=1}^{z}$ (t_i is a piece of the original text) and the pre-manually designed text patterns $\mathcal{P} = \{P_j\}_{j=1}^{m}$, we use $P_j(t_i)$ to denote the patterned sentence of t_i with P_j. This process can be formulated as:

$$\begin{cases} t_i = \underline{\underline{I\ really\ enjoy\ the\ food\ here.}},\ t_i \in \mathcal{T} \\ P_j(t_i) = It\ is\ [MASK],\ \underline{\underline{I\ really\ enjoy\ the\ food\ here.}}, P_j \in \mathcal{P}. \end{cases} \tag{1}$$

The goal of acquiring verbalizers is to select a suitable word w, i.e., the verbalizer, from the vocabulary set $\mathcal{V}$ to replace the $[MASK]$ token in the patterned sentence $P_j(t_i)$ in Eq. (1). In addition, we can evaluate the quality of the verbalizers based on the performance of the PET model in the downstream tasks. Given a pre-trained text classification model $\mathcal{M}$, the classification process can be formulated as $\mathcal{M}(t, L^M) \rightarrow l$, where $l \in L^M$ consisting of M different labels. Therefore, the accuracy of text classification can reflect the effectiveness of verbalizers to some extent with:

$$\min_{v \in \mathcal{V}} \frac{1}{z} \frac{1}{m} \sum_{i=1}^{z} \sum_{j=1}^{m} \mathbb{1}\{\mathcal{G}(t_i) - \mathcal{M}(P_j(t_i), v)\}, \tag{2}$$

where $\mathcal{G}(t_i)$ denotes the ground-truth label for the original text t_i, v is a verbalizer selected from the vocabulary set $\mathcal{V}$, $\mathbb{1}(\cdot,\cdot)$ is an indicator function that outputs 0 when two inputs belong to the same label, otherwise outputs 1. Then, the task studied in this paper is to acquire the verbalizer for each label automatically, which can be reformulated as follows. Given a classification task with the label set L^M, verbalizer acquisition aims to select a verbalizer $V(l_i)$ with:

$$V(l_i) \leftarrow \mathcal{S}(\mathcal{V}, l_i), \quad l_i \in L^M, \tag{3}$$

where $\mathcal{S}$ is a verbalizer acquire function that retrieves a word from the vocabulary $\mathcal{V}$ for a given label l_i.

To deal with the aforementioned task, we propose a **m**etric **se**ntiment **l**earning **f**ramework (MSeLF) based on **metric sentiment learning**. We plot the framework in Fig. 2. Clearly, MSeLF mainly consists of the **contrast map learning (CML)** component and the **equal-gradient verbalizer acquisition (EVA)** component, which will be detailed in the following sections, respectively.

3.2 Contrast map learning

The CML component aims to create a new antonym-aware embedding space so as to enlarge the distance between the embeddings of antonyms.

First, we construct a training set $\mathcal{T}_{map}$ with an external knowledge base of synonyms and antonyms provided by WordNet [3]. For a specific word x_i, if its synonym x_i^s and antonym x_i^a both appear in the original dataset $\mathcal{T}$, they will be included in $\mathcal{T}_{map}$ as a triplet $\{x_i, x_i^s, x_i^a\}$, while the rest of the words are included in the unmapped set $\mathcal{T}_{other}$. This process can be formulated as follows:

$$\begin{cases} \mathcal{T}_{map} = \cup_{i=1}^{n} x_i \quad s.t. \{x_i, x_i^a, x_i^s\} \in \mathcal{T} \\ \mathcal{T}_{other} = \cup_{i=1}^{n} x_i \quad s.t. \{x_i, x_i^a, x_i^s\} \notin \mathcal{T}. \end{cases} \tag{4}$$

According to the purpose of building a contrast mapping for an antonym-aware embedding space, we can infer a distance relationship in the embedding space. That is, a word should stay closer with its synonyms than with its antonyms. We formulate it as:

$$dist(x_i, x_i^s) < dist(x_i, x_i^a), \tag{5}$$

where $dist(\cdot,\cdot)$ is the distance score, which is evaluated by a negative cosine similarity between two vectors. For instance, given two inputs a and b, we can get:

$$dist(a, b) = -\frac{a \cdot b}{||a||_2^2 \cdot ||b||_2^2}, \qquad -1 \leq dist(a, b) \leq 1. \tag{6}$$

As existing approaches do not propose an effective way to improve the training efficiency of the contrast mapping with a few samples using both synonyms and antonyms, we adopt the structure

[3] https://wordnet.princeton.edu/

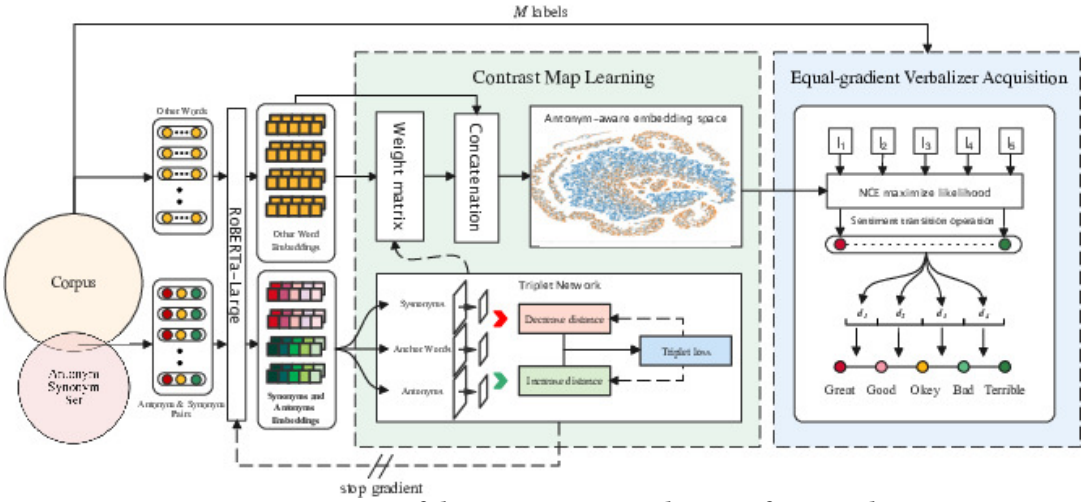


Figure 2: Overview of the metric sentiment learning framework.

of a triplet network composed of three fully connected networks with sharing weights to ensure that both synonyms and antonyms are involved in training. We present the Triplet Network in Fig. 3. Here, we input a triplet of an anchor word x and its synonym as well as antonym into the network. We take the synonym x^s as a positive sample and the antonym x^a as a negative sample. A loss function is designed to update the weights of the fully-connected network.

In detail, the outputs of the MLP network in the triplet networks can be formulated as:

$$e(x) = ReLu(W_b x + b), \tag{7}$$

where W_b and b are the weight matrix and the bias in the multi-layer perceptions layer, respectively. Here, the original input word embeddings are directly obtained from RoBERTa-Large [21], which are not updated throughout the training process to reduce the computational effort. By doing so, we can obtained the respective embeddings of the anchor word, synonym and antonym denoted as $e(x_i)$, $e(x_i^s)$ and $e(x_i^a)$.

Regarding the triplet ranking loss, following the idea of a triplet network [11] which favors a small distance between pairs of examples with the same label and a large distance for pairs with a different label, we formulate the training loss $\mathcal{L}_{map}$ for the contrast map learning component as:

$$\mathcal{L}_{map} = \sum_{\{x_i, x_i^s, x_i^a\} \in \mathcal{T}_{map}} \max\{0,\ \alpha + dist(e(x_i), e(x_i^s)) - dist(e(x_i), e(x_i^a))\} \quad s.t. \alpha > 0, \tag{8}$$

where α is a fixed threshold and the cosine similarity is applied to measure the distance.

To generate the embeddings of all words in $\mathcal{T}$ by CML, upon the triplet network, we multiply the weight matrix W_b in the triplet network with the initial word embedding matrix $E_{original}$ obtained from RoBERT-large, which is then concatenated with $E_{original}$ to produce the word embedding matrix E_{new}, i.e.,

$$E_{new} = E_{original} * W_b \oplus E_{original}. \tag{9}$$

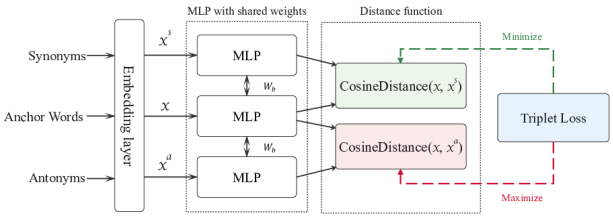


Figure 3: Overview of the Triplet Network for extracting the synonym-antonym distinctions.

3.3 Equal-gradient verbalizer acquisition

Motivated by the human understanding process of sentiment shift, the EVA module is used to obtain the representation words for each label based on the embeddings generated by CML.

Recall the situation where the verbalizer works, given a text $t_i \in \mathcal{T}$, the selection of label $l_i \in L^M$ is transformed into a selection of the verbalizer $v_l \in \mathcal{V}$. This transformation process can be formally expressed as:

$$Q(l_i|t_i) = \frac{expP^M(v_l|P_j(t_i))}{\sum_{k=1}^{|\mathcal{V}|} expP^M(v_k|P_j(t_i))} \tag{10}$$

where P^M is a conditional probability distribution, Q is the probability of generating label l_i given t_i, and P_j refers to a pattern from $\mathcal{P}$. We then use $\mathcal{T}_{l_i}$ to denote the text with label l_i. A natural criterion for the suitability of the verbalizer v_l is to calculate its likelihood to the training data $\mathcal{T}$, resulting in a maximum likelihood estimation:

$$v_l \leftarrow \arg\max_{v_i \in \mathcal{V}} \prod_{t_i \in \mathcal{T}_{l_i}} Q_{(P_j, v_i)}(l_i|t_i). \tag{11}$$

However, finding an optimal verbalizer by iterating $\mathcal{V}$ is difficult to implement and leads to an extremely high computation overhead. To overcome this problem, we apply the idea of Noise Contrastive Estimation (NCE) [25] to reform the multi-classification problem into a binary classification problem. To this end, we group the samples according to the corresponding labels, e.g., $\{l_1, l_2, \cdots, l_M\}$, and

denote the group with label l_i as the target class containing positive samples and the other groups as the noise classes containing negative samples. The reformed dataset is denoted as $\mathcal{T}_{\hat{l}}$. We use $\hat{l}$ to denote the label of samples in this circumstance:

$$\hat{l} = \begin{cases} 1, & if \ \ l = l_i \\ 0, & otherwise. \end{cases} \tag{12}$$

This reforming process creates a label imbalance, i.e., $\mathcal{T}_{\hat{l}}$ contains (M-1) times negative samples as positive ones. We address this imbalance by adding a factor s as:

$$s(\hat{l}) = \begin{cases} 1, & if \ \ \hat{l} = 1 \\ n_y/(|\mathcal{T}_{\hat{l}}| - n_y), & otherwise, \end{cases} \tag{13}$$

where n_y is the number of samples in $\mathcal{T}_{\hat{l}}$ with label $\hat{l} = 1$. This factor is similar to the imbalance solution suggested by Lee et al. [16] for the multi-class classification task with the support vector machines. We then reformulate the purpose of maximizing the likelihood $Q_{(P_j,v_i)}(l_i|t_i)$ in Eq. (11) as the aim of minimizing the cross entropy $L_{CE}(\mathcal{T}_{\hat{l}}, v_i)$ between the ground truth label $\hat{l}_i$ and the model prediction $q_{(P_j,v_i)}$ as:

$$L_{CE}(\mathcal{T}_{\hat{l}}, v_i) = -\sum_{(t_i,\hat{l}_i)\in\mathcal{T}_{\hat{l}}} s(\hat{l}_i) \cdot \log Q_{(P_j,v_i)}(\hat{l}_i|t_i). \tag{14}$$

Yet, as the size of vocabulary $\mathcal{V}$ is quite large, the model predicts the conditional probability distribution of $\hat{l} = 0$ given t_i is always close to 1 and its logarithm is close to 0, indicating that the negative samples contribute nearly nothing to the cross-entropy loss in Eq. (14). We solve this problem by considering the likelihood ratio of the conditional probability distribution like:

$$L_{LR}(\mathcal{T}_{\hat{l}}, v_i) = -\sum_{(t_i,\hat{l}_i)\in\mathcal{T}_{\hat{l}}} s(\hat{l}_i) \cdot \log \frac{Q_{(P_j,v_i)}(\hat{l}_i|t_i)}{Q_{(P_j,v_i)}(1-\hat{l}_i|t_i)}. \tag{15}$$

Note that we denote the groups with label l_1 or l_M as two extreme classes and other groups as intermediate classes. After obtaining the verbalizer for one of the extreme classes by Eq. (15), we use the antonym list to find its antonyms as candidates. The same likelihood ratio loss is then used to find the most suitable verbalizer for the other extreme class.

After we obtain the verbalizers for these two extreme classes v_1 and v_M, we use the previously mentioned sentiment transition operation to select the verbalizers for the intermediate classes. In particular, we use the difference of the extreme class verbalizers in the embedding space as the sentiment transition vector $\vec{g}_s$ like:

$$\vec{g}_s = v_M - v_1. \tag{16}$$

Then, we calculate the sentiment gradient, i.e., the ideal difference between each class, by a division operation as:

$$\Delta_s = \frac{1}{M-1} \times \vec{g}_s \tag{17}$$

Finally, we apply the sentiment gradient to the verbalizers of the extreme classes to obtain the verbalizers of intermediate classes.

The detailed process of EVA can refer to Algorithm 1. We first use the likelihood ratio loss in Eq. (15) to select an antonym pair as verbalizers for two extreme classes as line 1 and 2. Then, we compute the difference of the embeddings of two verbalizers as the sentiment transition vector as shown in line 3. We then divide the sentiment transition vector by $M-1$ to produce the sentiment gradient as a total of M classes are contained in the dataset as line 4 and 5. For the j-th intermediate class, we add $j-1$ sentiment gradients to the embeddings of the verbalizer of the first extreme class, which is then used as the prototype of the j-th class [45]. Finally, we calculate the distances between the words in the vocabulary and this generated prototype, and pick up the word with the smallest distance as the verbalizer of the j-th class as shown in lines 6 and 7.

Algorithm 1 Equal-gradient verbalizer acquisition

Input: M is the number of classes in the dataset, $\mathcal{V}$ is the vocabulary. $prototype()$ denotes the word prototype for a specific class [45]. L_{LR} denotes the likelihood ratio loss in Eq. (15). $antonym(w)$ denotes the set of antonyms of w.
Output: Verbalizers of all categories $\{v_1, \cdots, v_M\}$.
1: Obtain verbalizer for v_1: $v_1 \leftarrow \arg\min_{w\in\mathcal{V}} L_{LR}$;
2: Obtain verbalizer for v_M: $v_M \leftarrow \arg\min_{w\in antonym(v_1)} L_{LR}$;
3: Obtain the sentiment transition vector: $\vec{g}_s = v_M - v_1$;
4: **for** j *in* $(M-1)$ **do**
5: $\quad prototype(v_j) \Leftarrow v_1 + \frac{j-1}{M-1} \times \vec{g}_s$;
6: $\quad$ **for** $word \in \mathcal{V}$ **do**
7: $\quad\quad v_j \Leftarrow \arg\min_{word} dist(word, prototype(v_j))$;
8: $\quad$ **end for**
9: **end for**
10: **return** $\{v_2, \cdots, v_{M-1}\}$.

4 EXPERIMENTAL SETUP

In this section, we first present the datasets used for experiments in Sec. 4.1. Then, we discuss the research questions in Sec. 4.2. Finally, we summarize the baselines and our proposals in Sec. 4.3.

4.1 Datasets

For the data used for pretraining CML, we employ the same enriching method [39] on several publicly available sources [4] to expand the word list with a large number of synonyms and antonyms. The expanded word list consists of 11,109 antonym pairs and 68,971 synonym pairs. To cover the synonyms and antonyms in the validation sets, we reduce the redundant word pairs in the expanded word list, resulting in 4,640 antonym pairs and 22,736 synonym pairs. These retained word pairs can be further partitioned into 23,494 pairs in the training set and 3,882 pairs in the test set.

We evaluate the performance of the state-of-the-art baselines and our proposed model on two datasets, i.e., Yelp-Reviews [5] and Amazon-Reviews [6], which are commonly used for sentiment classification. Yelp-review collects data from the restaurant industry and Amazon-review refers to user opinions from a number of sectors including books, movies, sports and health. Table 1 details the rating distribution in the dataset.

To construct a low-resource scenario, we randomly obtain a series of training sets $\mathcal{T}$ consisting of t samples from the dataset, i.e., $t \in \{10, 50, 100, 1000\}$, leading to four individual sub datasets.

[4] https://github.com/ec2604/Antonym-Detection, https://wordnet.princeton.edu/
[5] https://www.kaggle.com/yelp-dataset/yelp-dataset
[6] https://www.kaggle.com/snap/amazon-fine-food-reviews

Table 1: Statistics of the datasets used in our experiments.

Dataset	Yelp-Review	Amazon-Review
label = "1"	14.6%	14.2%
label = "2"	8.2%	5.2%
label = "3"	10.7%	7.5%
label = "4"	22.2%	14.1%
label = "5"	44.2%	63.9%
# Domains	5	24
# Total samples	8,635,403	568,454

Similar to the definition of episodic learning [46], we construct low-resource scenarios by limiting the number of sample instances in each category. This construction is also more in line with real-life situations where a good or service can be counted correctly within limited user feedback. In each sub dataset, we draw an equal number of instances for each particular category. For the patterns used for evaluations, we employ the same operation with PET [43] to reformulate the inputs.

4.2 Research questions and configurations

4.2.1 Research questions. We list several research questions to guide the experiments and verify the effectiveness of our proposal.

RQ1: Does our proposed model improve the performance of sentiment classification in the deficient-data scenario compared to the state-of-the-art baselines?

RQ2: How does our model perform under different settings of data availability?

RQ3: Can the contrasting mapping weights effectively separate the antonymous words in the reformed embedding space?

RQ4: How does the text length affect the model performance for sentiment classification especially in the low-resource scenario?

RQ5: Which component contributes more to improving the model performance, CML or EVA?

4.2.2 Model configurations. We directly use RoBERTa-Large [21] to obtain the initial word embeddings of synonyms as well as antonyms, which are then applied to the contrasting mapping construction module. We use a triplet network consisting of three parallel fully connected networks, which share two linear transformation layers with weights of 1024×500 and 500×150, respectively. In the training process, following [39], we set the batch size to 64 and adopt ADAM [33] with a learning rate $1e^{-3}$ to optimize our model. In addition, we set the dropout rate to 0.2, and the number of iterations to 200.

For training PET, following [43], we refer to RoBERTa as the encoder and apply the language modeling as an auxiliary task with a weight term of 0.999. In addition, we set the learning rate to $1e^{-5}$, the weight decay to 0.01, the ADAM epsilon to $1e^{-8}$, the temperature to 2, the training count to 3, the max length to 256, and the training batch size to 4, respectively.

4.3 Model summary

For all models discussed, we employ the same word encoder, i.e., the RoBERTa-large encoder [21], with the intention to fairly compare their performance on different size of training sets. Here, we list a series of state-of-the-art baselines for comparisons with our proposal in this paper:

Supervised [43]: A regular supervised learning model that add a sequence classification layer to the language model.

PET+Random [43]: We randomly select 5 sets of words for each label from the training set and invert their positions in the training procedure to form 10 sets of verbalizers.

PET+AVS [43]: AVS first assigns each word in $\mathcal{V}$ to each label l, and then computes the similarity score of each word to the sample belonging to l using a pre-trained language model $\mathcal{M}$.

PET+Petal [42]: Petal is an improved method for automatic extraction of verbalizers based on AVS. Petal uses the likelihood ratios to calculate the scores of individual verbalizers.

PET+MSeLF: MSeLF proposed in this paper consists of two major components, i.e., the contrast mapping learning (CML) module and the equal-gradient verbalizer acquisition (EVA) module.

PET+Manual: We follow the verbalizer set manually in PET. Their filling makes the pattern-reconstructed sentences readable and correctly expresses the original sentiment. Such manually designed method presents the upper bounds of auto verbalizers.

Next, we list the model variants proposed in this paper upon PET for further discussions. Since the CML module contained in our proposal is a flexible plug-in to the above-mentioned verbalizer baselines, we would like to investigate its performance when it's incorporated into other baselines.

PET+MSeLF$_{[Overlap]}$: We remove the entire CML module from MSeLF and apply the EVA module directly to the word embeddings that are not fine-tuned.

PET+MSeLF$_{[Random]}$: A random verbalizer acquisition strategy proposed in [42] is incorporated with the CML module.

PET+MSeLF$_{[AVS]}$: A verbalizer acquisition strategy AVS proposed in [43] is incorporated with the CML module.

PET+MSeLF$_{[Petal]}$: A verbalizer acquisition strategy Petal proposed in [42] is incorporated with the CML module.

5 RESULTS AND ANALYSIS

5.1 Overall performance

To answer **RQ1**, we evaluate the performance of our proposal based on the metric sentiment learning framework (MSeLF) and five competitive baselines for the downstream text classification task on two public datasets. We present the results of involved models for discussions in a few-shot setting with sample size $|\mathcal{T}| \in \{10, 50, 100, 1000\}$ in Table 2, respectively.

Generally, comparing the model performance on the Yelp-Review dataset against that on the Amazon-Review dataset, we can observe that the models mostly perform relatively better on the former than on the latter dataset. It could be explained by the fact that Amazon-Review contains more domains than Yelp-Review, resulting in difficulties for models to classify the text correctly. In particular, "PET+Manual" performs the best among the models, with a noticeable accuracy improvement over other four baselines. For instance, on the Yelp-Review dataset with $|\mathcal{T}| = 10$, "PET+Manual" presents an improvement of 24.4%, 29.7%, 5.7%, 2.6% and 3% in terms of accuracy against the "Supervised", "PET+Random", "PET+AVS", "PET+Petal" and "PET+MSeLF" models, respectively, while shows a corresponding 3%, 5.9%, 1.1%, 1% and 0.4% on the Amazon-Review dataset with $|\mathcal{T}| = 1000$. This indicates that the closer the manual verbalizer keeps to the natural language and human cognition, the

Table 2: Classification accuracy of six classification models. The results of the best algorithmic performer and baseline in each column are bolded and underlined, respectively.

Models	$\|\mathcal{T}\| = 10$		$\|\mathcal{T}\| = 50$		$\|\mathcal{T}\| = 100$		$\|\mathcal{T}\| = 1000$	
	Yelp	Amazon	Yelp	Amazon	Yelp	Amazon	Yelp	Amazon
Supervised	21.5 ± 1.6	21.4 ± 0.9	44.8 ± 0.7	43.2 ± 0.6	50.0 ± 0.1	49.5 ± 0.2	60.2 ± 0.5	58.2 ± 0.4
PET+Random	16.2 ± 1.2	19.3 ± 0.5	31.2 ± 0.1	29.5 ± 1.5	46.3 ± 0.9	45.2 ± 0.7	54.0 ± 1.1	55.3 ± 0.8
PET+AVS	40.2 ± 1.6	39.8 ± 1.9	52.3 ± 0.4	49.9 ± 0.2	56.1 ± 0.4	53.9 ± 0.2	61.1 ± 0.2	60.1 ± 0.4
PET+Petal	**43.3 ± 0.4**	**41.1 ± 0.6**	53.5 ± 0.6	50.8 ± 0.1	57.3 ± 1.4	54.4 ± 0.6	61.7 ± 0.5	60.2 ± 1.7
PET+MSeLF (Ours)	42.9 ± 0.3	40.8 ± 0.6	**53.8 ± 4.6**	**51.1 ± 0.6**	**58.2 ± 2.2**	**55.4 ± 0.6**	**63.2 ± 0.3**	**60.8 ± 1.4**
PET+Manual	45.9 ± 0.1	46.1 ± 1.3	59.2 ± 0.7	54.5 ± 1.3	61.2 ± 0.9	57.9 ± 0.5	63.4 ± 1.3	61.2 ± 1.2

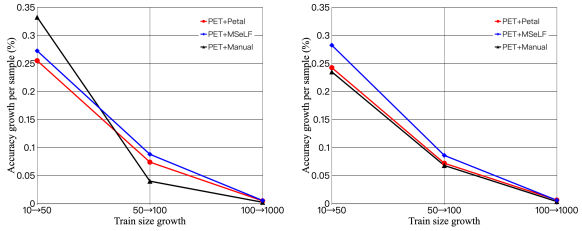
better performance the model can achieve. Thus, we consider the "PET+Manual" approach as an upper bound of all models proposed in this paper.

To validate the superiority of the plug-in modules like Petal and MSeLF rather than PET itself, we present the results of the "Supervised" and "PET+Random" models. It is clear that "PET+Random" performs worse than "Supervised" on both datasets for all cases. For instance, on the Yelp-Review dataset with $|\mathcal{T}| = 10, 50, 100, 1000$, "PET+Random" shows a corresponding 5.3%, 13.6%, 3.7% and 5.8% decrease in terms of accuracy against the "Supervised" model. In addition, "PET+Petal" presents an obvious superiority over other three baselines, i.e., "Supervised", "PET+Random" and "PET+AVS". For example, on the Yelp-Review dataset with $|\mathcal{T}| = 10$, "PET+Petal" shows a corresponding 21.8%, 27.1% and 3.1% improvements in terms of accuracy against the baselines "Supervised", "PET+Random" and "PET+AVS", respectively. This means Petal is able to find the discriminatory words for each category in the limited input word list as a verbalizer. Hence, in the following sections, we focus on the "PET+Manual" and "PET+Petal" models for comparisons with our proposals.

Regarding our proposal, i.e., "PET+MSeLF", it generally outperforms "PET+Petal" on both datasets with $|\mathcal{T}| = 50, 100, 1000$ and still leaves a gap to "PET+Manual". For instance, "PET+MSeLF" presents 1.5% and 0.6% improvement in terms of accuracy over "PET+Petal" on Yelp-Review and Amazon-Review with $|\mathcal{T}| = 1000$, respectively. It can be explained by the fact that MSeLF can extract the verbalizers that can better represent labels of both extreme classes and intermediate classes than Petal. It's worth noting that "PET+MSeLF" underperforms "PET+Petal" on both datasets with $|\mathcal{T}| = 10$. It could be attributed to the fact that for case with $|\mathcal{T}| = 10$, a limited number of samples lead to a very small vocabulary. In such situation, Petal is able to select a relatively suitable verbalizer in a relatively small vocabulary, whereas MSeLF is less adaptable than Petal in such low-resource situations. In other words, "PET+MSeLF" prefers to a situation with a relatively large number of training samples.

5.2 Effect of data volume

To answer **RQ2**, for a particular investigation on how the training sample size affects the model performance, we plot the accuracy growth rates of our proposal "PET+MEsLF" and the outstanding baselines, i.e., "PET+Manual" and "PET+Petal" on Yelp-Review and Amazon-Review in Fig. 4. For straightforwardly presenting the



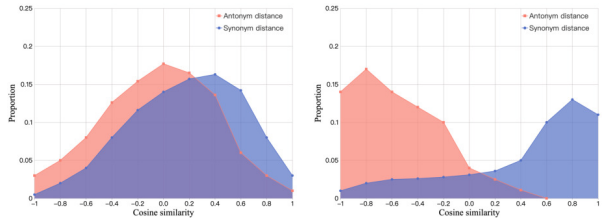
(a) Yelp-Review. **(b) Amazon-Review.**

Figure 4: Average accuracy growth per sample.

relation between the model performance and the sample size, we assume that the accuracy improvement is produced by the number of samples gradually and thus can be allocated to each individual sample. Accordingly, we define the growth rate as the value of the accuracy increase divided by the amount of added instances. Clearly, the overall performance of the models discussed maintains a steady upward trend as the amount of training data increases. This phenomenon suggests that increasing the amount of data can reduce the overfitting issue of the corresponding models and improve the differentiation ability. Another obvious observation is that the increase rate of accuracy goes down as the size of the training data raises. It can be explained by the fact that as the sample size grows, the input text becomes complex, making it difficult for the model to fit the full data.

In detail, we zoom in the accuracy growth rate on the Yelp-Review dataset first. It can be observed that "PET+Manual" has the highest growth rate as the training set size grows from 10 to 50, with a 0.08% and 0.06% advantage over that of "PET+Petal" and "PET+MSeLF", respectively. In contrast, "PET+Manual" presents the lowest growth rate when the training set size grows from 50 to 100, with a decrease of 0.03% and 0.48% against that of "PET+Petal" and "PET+MSeLF", respectively. Differently, on Fig. 4b, "PET+Manual" shows the smallest accuracy growth rate when the dataset size grows from 10 to 50 compared with "PET+Petal" and "PET+MSeLF". This same phenomenon of the accuracy growth trend for all models on Amazon-Review and Yelp-Review indicates that the manual verbalizer approach allows the model to converge faster than other two algorithmic models.

Next, we focus on comparing the growth rates of "PET+Petal" and "PET+MSeLF". It can be observed that the growth rate of "PET+MSeLF" is consistently higher than that of "PET+Petal" on



(a) Antonym distance and synonym distance before CML. (b) Antonym distance and synonym distance after CML.

Figure 5: Distributions of antonyms and synonyms.

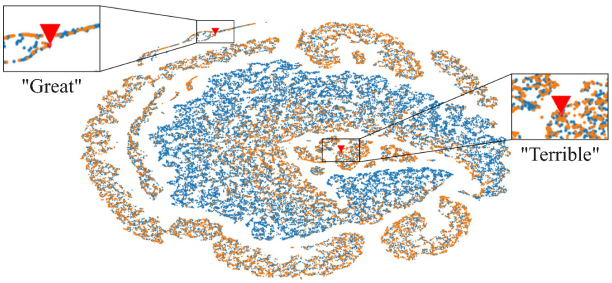


Figure 6: Antonym-aware embedding space.

both datasets. For example, on Yelp-Review, an improvement of 0.04% and 0.02% in terms of the growth rate is observed by comparing "PET+MSeLF" against "PET+Petal" when the data size grows from 10 to 50 and 50 to 100, respectively. This indicates that our proposal "PET+MSeLF" can make better use of the newly added samples than "PET+Petal", resulting in a superiority of "PET+MSeLF" over "PET+Petal" for the downstream classification task when more training data is given.

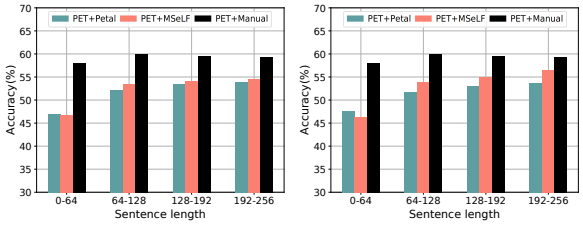
5.3 Embedding visualization

To answer **RQ3**, we plot the ratios of synonyms and antonyms according to their specific cosine similarity distance in Fig. 5a and Fig. 5b, respectively. Moreover, we visualize the new embedding space created by CML in Fig. 6 and plot a particular pair of antonym words, i.e., "Great" and "Terrible", where the points represent the distribution of word embeddings after a dimensionality reduction scheme know as TSNE [37].

Clearly, Fig. 5a shows the distance between antonyms and synonyms before CML is performed while Fig. 5b presents the corresponding distances after CML is performed. From Fig. 5a, we can notice that the embeddings of synonyms and antonyms before the CML training keep close to each other as the majority of pairs are associated with a relatively low cosine similarity score near zero. However, these pairs can be well separated after CML is implemented as the cosine similarity scores keep close to -1 for antonyms and +1 for synonyms, respectively, as shown in Fig. 5b. The similar phenomenon can be observed by a specific case in Fig. 6, where the antonym pair in red, i.e., "Great" and "Terrible", is well separated. Thus, we would like to conclude that CML has the following characteristics:

(1) CML can indeed enlarge the distance of antonym pairs. At the same time, the distance of synonym pairs can be shortened. In other words, CML can well explore the semantics of the antonyms and synonyms.
(2) Other words excluding the synonyms and the antonyms will not be distributed by CML. In other words, they can maintain their original relative semantics in the new embedding space, which makes sense for the downstream tasks.

From Fig. 5b, we can also find that for some cases, the distance is still small for antonyms. However, such errors will not have a noticeable impact on the final results of sentiment metric learning. It could be due to the fact that the unsuccessful separation of antonyms will be filtered out during the calculation process of the likelihood ratio.



(a) Accuracy on Yelp-Review. (b) Accuracy on Amazon-Review.

Figure 7: Model performance on cases with different text length.

5.4 Impact of sentence length

Next, to answer **RQ4**, for simplicity, we implement our experiments in the setting of $|\mathcal{T}| = 50$. We group all input texts for testing in Yelp-Review and Amazon-Review according to their length, so as to analyze the impact of the text length on the performance of our proposal as well as the baselines. Specifically, considering the limitation of input text length imposed by the PET framework, i.e., the max length is set to 256, we group the input sentences into four groups, i.e., (0, 64), [64, 128), [128, 192) and [192, 256). The results of the corresponding models for the sentiment classification task on texts with different lengths are plotted in Fig. 7. Here, we only present the results of the best baseline as well as the approach with the manual verbalizer for comparisons as they show the outstanding performance in Sec. 5.1.

As shown in Fig. 7, we can observe that the manual setting is relatively less sensitive to the length of the input text than other models, whereas the performance of the "PET+Petal" model and the "PET+MSeLF" model consistently goes up along with an increasing length of the input text. This could be due to the fact that the manually generated verbalizer contains the human-reserved knowledge, and accordingly is unaffected by the input data. However, the automatic verbalizer acquisition methods select the appropriate words in the vocabulary of a given dataset, where the long texts will contribute a relatively large vocabulary under a fixed size of samples, leading to difficulties for verbalizer selection. In addition, our proposal, i.e., "PET+MSeLF", seems more sensitive to the text length than "PET+Petal". For instance, "PET+MSeLF" wins the competition against "PET+Petal" for cases in the group [64, 128) and loses the comparison for cases in the group (0, 64). It can be explained by the fact that the longer text inputs contain

Table 3: Ablation study results of MSeLF on Yelp-Review and Amazon-Review with different amounts of data. The results of the best performer in each column are bolded. The biggest drop in each column is appended ↓.

Model change	$\|\mathcal{T}\|$=10		$\|\mathcal{T}\|$=50		$\|\mathcal{T}\|$=100		$\|\mathcal{T}\|$=1000	
	Yelp	Amazon	Yelp	Amazon	Yelp	Amazon	Yelp	Amazon
PET+MSeLF$_{[Overlap]}$	10.2 ↓	12.3 ↓	26.8 ↓	24.2 ↓	33.5 ↓	34.3 ↓	43.4 ↓	42.9 ↓
PET+MSeLF$_{[Random]}$	16.2	19.4	30.4	28.8	49.5	48.9	53.2	54.0
PET+MSeLF$_{[AVS]}$	41.2	40.8	52.4	51.3	57.0	52.7	62.0	60.3
PET+MSeLF$_{[Petal]}$	**43.4**	**41.3**	53.6	50.6	59.4	54.8	62.1	60.4
PET+MSeLF (Ours)	42.9	40.8	**53.8**	**51.1**	**58.2**	**55.4**	**63.2**	**60.8**

more complex semantic information that can interfere the matching degree calculation process in Petal. In contrast, MSeLF only uses the matching method to select the verbalizers for categories containing the opposite contexts where the sentiment is evident. In addition, MSeLF applies the metric method to select the verbalizers for the intermediate classes, enabling the interference of useless information in the text to be avoided.

5.5 Ablation study

To answer **RQ5**, we conduct an ablation study to validate the utility of the CML module and EVA module in MSeLF. In particular, we remove the CML module to evaluate the influence of the antonym-aware embeddings and replace EVA with other verbalizer acquisition schemes, i.e., Random, AVS, and Petal upon CML to evaluate the effectiveness of EVA. We present the results of the ablation study in Table 3.

By comparing "PET+MSeLF" and "PET+MSeLF$_{[Overlap]}$", we notice that if CML is removed, it will cause a significant reduction in the classification capability. In particular, a decrease of 32.7%, 27%, 24.7% and 17.9% in terms of accuracy is returned when comparing "PET+MSeLF$_{[Overlap]}$" against "PET+MSeLF" on Yelp-Review with size $|\mathcal{T}| = 10, 50, 100, 1000$, respectively. It is because in the absence of CML, the verbalizers of the two extreme classes selected by the EVA calculation are very close in the embedding space, resulting in the verbalizers of the intermediate classes staying semantically close as well and failing to generate representative labels.

Next, we take the CML module as a plug-in upon the representative baselines, i.e., "PET+Random", "PET+AVS" and "PET+Petal" , which then generates the corresponding "PET+MSeLF$_{[Random]}$" , "PET+MSeLF$_{[AVS]}$" and "PET+MSeLF$_{[Petal]}$" models, respectively. We can find that the performance of the integrated models generally outperform their original models. For instance, "PET+MSeLF$_{[Petal]}$" outperforms "PET+Petal" by 2.1% accuracy improvement while "PET+MSeLF$_{[AVS]}$" outperforms "PET+AVS" by 0.9% accuracy improvement on Yelp-Review with $|\mathcal{T}| = 10$. This demonstrates that the new word embeddings generated by CML can indeed improve the performance of other baselines, indicating that CML can be used as a flexible plug-in to complement other verbalizer acquisition methods. The exception is returned on the random selection, where the randomly selected verbalizer is completely unaffected by the changes, which means that the CML module can not benefit the verbalizer acquisition.

In addition, to examine the effectiveness of EVA, as shown in Table 3, "PET+MSeLF$_{[Petal]}$" slightly outperforms "PET+MSeLF" on Yelp-Review and Amazon-Review with $|\mathcal{T}| = 10$, presenting the same improvement of 0.5% in terms of accuracy, which is consistent with the findings in Sec. 5.1 as Petal is adaptive to very small sample sizes. However, "PET+MSeLF" outperforms "PET+MSeLF$_{[Petal]}$" on both datasets with $|\mathcal{T}| = 50, 100, 1000$. For instance, "PET+MSeLF" shows an improvement of 0.5%, 0.6%, 0.4% against "PET+MSeLF$_{[Petal]}$" on Amazon-Review with $|\mathcal{T}| = 50, 100, 1000$, respectively. This proves that the EVA module is able to make better use of the antonym-aware word embeddings produced by the CML module than other verbalizer acquisition models, thus can select verbalizers that are more capable of category differentiation.

6 CONCLUSION AND FUTURE WORK

Our work in this paper aims to generate the verbalizer for label representation. We introduce a novel sentiment metric learning framework (MSeLF) consisting of a contrast mapping learning (CML) module and an equal-gradient verbalizer acquisition (EVA) module. Specifically, CML learns a transformation matrix to project the initial word embeddings to the antonym-aware embeddings by enlarging the distance between the antonyms. After that, in the antonym-aware embedding space, EVA takes a pair of antonymous words as verbalizers for two opposite classes and then uses the sentiment transition operation to acquire verbalizers that can effectively represent the intermediate classes. Finally, MSeLF can generate correct verbalizers for the downstream text classification task in a few-shot setting.

As to future work, on the one hand, we plan to improve the word embeddings by a stream-based calibration method [17], as the word embeddings that are more in line with the true semantics can make the generated verbalizer more closely match the human understanding. In addition, we would like to implement this idea to the task of automatic verbalizer acquisition to further improve the quality of verbalizers. On the other hand, we plan to test our proposal on other datasets where the the labels with sentiment gradients are available.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China under No. 61702526, and the Postgraduate Scientific Research Innovation Project of Hunan Province. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Heike Adel and Hinrich Schütze. 2014. Using Mined Coreference Chains as a Resource for a Semantic Task. In *EMNLP*. 1447–1452.

[2] Muhammad Asif Ali, Yifang Sun, et al. 2019. Antonym-Synonym Classification Based on New Sub-Space Embeddings. In *AAAI*. 6204–6211.

[3] Marina Angelovska, Sina Sheikholeslami, et al. 2021. Siamese Neural Networks for Detecting Complementary Products. In *EACL*. 65–70.

[4] José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *AI* 240 (2016), 36–64.

[5] Lawrence Cayton. 2005. Algorithms for Manifold Learning. *UCSDT* 12, 1-17 (2005), 1.

[6] Jia-Ren Chang and Yong-Sheng Chen. 2018. Pyramid stereo matching network. In *CVPR*. 5410–5418.

[7] Xinyi Chen, Jingxian Xu, and Alex Wang. 2020. Label Representations in Modeling Classification as Text Generation. In *IJCNLP*. 160–164.

[8] Kyunghyun Cho, Bart van Merrienboer, et al. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*. 1724–1734.

[9] Zhen Dong, Su Jia, et al. 2017. Deep Manifold Learning of Symmetric Positive Definite Matrices with Application to Face Recognition. In *AAAI*. 4009–4015.

[10] Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. In *EMNLP*. 671–683.

[11] Elad Hoffer and Nir Ailon. 2015. Deep Metric Learning Using Triplet Network. In *SIMBAD*, Vol. 9370. 84–92.

[12] Hexiang Hu, Guang-Tong Zhou, et al. 2016. Learning Structured Inference Neural Networks with Label Relations. In *CVPR*. 2960–2968.

[13] Zhengbao Jiang, Frank F. Xu, et al. 2020. How Can We Know What Language Models Know. *TACL* 8 (2020), 423–438.

[14] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese Neural Networks for One-shot Image Recognition. In *ICML*, Vol. 2.

[15] Buon Kiong Lau, Jrgen Bach Andersen, et al. 2006. Impact of matching network on bandwidth of compact antenna arrays. *TAP* 54, 11 (2006), 3225–3238.

[16] Yoonkyung Lee, Yi Lin, and Grace Wahba. 2004. Multicategory Support Vector Machines: Theory and Application to The Classification of Microarray Data and Satellite Radiance data. *JASA* 99, 465 (2004), 67–81.

[17] Bohan Li, Hao Zhou, et al. 2020. On the Sentence Embeddings from Pre-trained Language Models. In *EMNLP*. 9119–9130.

[18] Dekang Lin, Shaojun Zhao, et al. 2003. Identifying Synonyms among Distributionally Similar Words. In *IJCAI*. 1492–1493.

[19] Jian Liu, Yubo Chen, et al. 2020. Event Extraction as Machine Reading Comprehension. In *EMNLP*. 1641–1651.

[20] Xiaoqian Liu, Fengyu Zhou, et al. 2020. Meta-Learning Based Prototype-relation Network for Few-shot Classification. *Neurocomputing* 383 (2020), 224–234.

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, et al. 2019. Roberta: A Robustly Optimized Bert Pretraining Approach. *CoRR* abs/1907.11692 (2019).

[22] Michael Loster, Ioannis K. Koumarelas, and Felix Naumann. 2021. Knowledge Transfer for Entity Resolution with Siamese Neural Networks. *ACM* 13, 1 (2021), 2:1–2:25.

[23] Tomás Mikolov, Kai Chen, et al. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.

[24] Tomás Mikolov, Ilya Sutskever, et al. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.

[25] Andriy Mnih and Yee Whye Teh. 2012. A Fast and Simple Algorithm for Training Neural Probabilistic Language Models. *CoRR* abs/1206.6426 (2012).

[26] Saif Mohammad and Peter Turney. 2010. Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *NAACL*. 26–34.

[27] Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating Distributional Lexical Contrast into Word Embeddings for Antonym-Synonym Distinction. In *ACL*. 454–459.

[28] Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Distinguishing Antonyms and Synonyms in a Pattern-based Neural Network. In *EACL*. 76–85.

[29] Rodrigo Nogueira, Zhiying Jiang, et al. 2020. Document Ranking with a Pretrained Sequence-to-Sequence Model. In *EMNLP*. 708–718.

[30] Haibo Pang, Qi Xuan, et al. 2021. Research on Target Tracking Algorithm Based on Siamese Neural Network. *MIS* 2021 (2021), 6645629:1–6645629:11.

[31] Fabio Petroni, Tim Rocktäschel, et al. 2019. Language Models as Knowledge Bases?. In *EMNLP*. 2463–2473.

[32] Nghia The Pham, Angeliki Lazaridou, and Marco Baroni. 2015. A Multitask Objective to Inject Lexical Contrast into Distributional Semantics. In *ACL*. 21–26.

[33] Timo Pukkala and Jari Miina. 1997. A Method for Stochastic Multiobjective Optimization of Stand Management. *FEM* 98, 2 (1997), 189–203.

[34] Yuanyuan Qiao, Yuewei Wu, et al. 2020. Siamese Neural Networks for User Identity Linkage Through Web Browsing. *TNNLS* 31, 8 (2020), 2741–2751.

[35] Alec Radford, Jeffrey Wu, et al. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI* 1, 8 (2019), 9.

[36] Colin Raffel, Noam Shazeer, et al. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *JMLR* 21 (2020), 140:1–140:67.

[37] Paulo E. Rauber, Alexandre X. Falcão, and Alexandru C. Telea. 2016. Visualizing Time-Dependent Data Using Dynamic t-SNE. In *ECV*. 73–77.

[38] Michael Roth and Sabine Schulte im Walde. 2014. Combining Word Patterns and Discourse Markers for Paradigmatic Relation Classification. In *ACL*. 524–530.

[39] Igor Samenko, Alexey Tikhonov, et al. 2020. Synonyms and Antonyms: Embedded Conflict. *CoRR* abs/2004.12835 (2020).

[40] Lawrence K. Saul and Sam T. Roweis. 2003. Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifold. *JMLR* 4 (2003), 119–155.

[41] Silke Scheible, Sabine Schulte im Walde, and Sylvia Springorum. 2013. Uncovering Distributional Differences between Synonyms and Antonyms in a Word Space Model. In *IJCNLP*. 489–497.

[42] Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically Identifying Words That Can Serve as Labels for Few-Shot Text Classification. In *COLING*. 5569–5578.

[43] Timo Schick and Hinrich Schütze. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *EACL*. 255–269.

[44] Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric Pattern Based Word Embeddings for Improved Word Similarity Prediction. In *CoNLL*. 258–267.

[45] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *NIPS*. 4077–4087.

[46] Flood Sung, Yongxin Yang, et al. 2018. Learning to Compare: Relation Network for Few-shot Learning. In *CVPR*. 1199–1208.

[47] Peter D. Turney and Patrick Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *JAIR* 37 (2010), 141–188.

[48] Oriol Vinyals, Charles Blundell, et al. 2016. Matching Networks for One Shot Learning. In *NIPS*. 3630–3638.

[49] Wenpeng Yin. 2020. Meta-learning for Few-shot Natural Language Processing: A Survey. *CoRR* abs/2007.09604 (2020).

[50] Sung Whan Yoon, Jun Seo, and Jaekyun Moon. 2019. TapNet: Neural Network Augmented with Task-Adaptive Projection for Few-Shot Learning. In *ICML*, Vol. 97. 7115–7123.

[51] Yaru Zhang, Yaqian Li, et al. 2021. Attention-guided Aggregation Stereo Matching Network. *IVC* 106 (2021), 104088.

[52] Jianming Zheng, Fei Cai, et al. 2019. Hierarchical Neural Representation for Document Classification. *Cogn. Comput.* 11, 2 (2019), 317–327.

[53] Jianming Zheng, Fei Cai, et al. 2020. Pre-train, Interact, Fine-tune: a novel interaction representation for text classification. *IPM* (2020), 102215.

[54] Jianming Zheng, Fei Cai, and Honghui Chen. 2020. Incorporating Scenario Knowledge into A Unified Fine-tuning Architecture for Event Representation. In *SIGIR*. ACM, 249–258.

[55] Jianming Zheng, Fei Cai, and Wanyu Chen others. 2021. Taxonomy-aware Learning for Few-Shot Event Detection. In *WWW*. ACM, 3546–3557.