Confidence-Driven Contrastive Learning for Document Classification without Annotated Data

Zhewei XU†a), Nonmember and Mizuho IWAHARA†, Member

SUMMARY Data sparsity has always been a problem in document classification, for which semi-supervised learning and few-shot learning are studied. An even more extreme scenario is to classify documents without any annotated data, but using only category names. In this paper, we introduce a nearest neighbor search-based method Con2Class to tackle this tough task. We intend to produce embeddings for predefined categories and predict category embeddings for all the unlabeled documents in a unified embedding space, such that categories can be easily assigned by searching the nearest predefined category in the embedding space. To achieve this, we propose confidence-driven contrastive learning, in which prompt-based templates are designed and MLM-maintained contrastive loss is newly proposed to finetune a pretrained language model for embedding production. To deal with the issue that no annotated data is available to validate the classification model, we introduce confidence factor to estimate the classification ability by evaluating the prediction confidence. The language model having the highest confidence factor is used to produce embeddings for similarity evaluation. Pseudo labels are then assigned by language model having the highest confidence factor is used to produce embeddings for similarity evaluation. Pseudo labels are then assigned by language model having the highest confidence factor is used to produce the classification ability by evaluating the prediction confidence. The final prediction. Our experiments on five representative datasets demonstrate the superiority of our proposed method over the existing approaches.

key words: Text classification, Semantic similarity, Contrastive learning, Self-training

1. Introduction

Document classification occupies an important role in information retrieval, which is widely applied in the fields like document management, news categorization and sentiment analysis. However, data sparsity is always an issue frequently happening in document classification. To meet the challenge of data deficiency, semi-supervised learning [1] and few-shot learning [2, 3] have been studied, in which model training is supervised by a small number of labeled samples. An even more extreme scenario is using only document contents and category names for classification, without any annotated data [4, 5].

To handle this tough task, several works leverage word-level features for classification [4, 6]. However, concrete words can hardly express accurate semantic information, increasing the risk of misclassification. Another approach is categorizing through comparing cosine similarities between document and category embeddings, which is also known as nearest neighbor classification [5, 7, 8, 9]. However, document embeddings do not necessarily locate around the embedding of their correct category in the embedding space, for which aligning document and category embeddings to bridge over semantic gaps is usually necessary, requiring extra manual/computing costs.

In this paper, we intend to predict category embeddings for unlabeled documents and compare these predicted category embeddings with predefined category embeddings for classification, such that there is no need to bridge over the semantic gap between documents and predefined category names. To this end, we propose Con2Class, in which a pretrained language model (PLM) is leveraged to produce two types of embeddings, one from predefined category names and the other from unlabeled documents. These category embeddings form a unified space, on which the category of one document can be easily classified by searching the nearest predefined category, without the need of alignment. Specifically, we enter category name-contained documents to the PLM to produce contextualized embeddings of predefined categories from the last hidden layer of the predefined category name. Meanwhile, we utilize manually designed prompt-included templates to guide the PLM to predict category embeddings for all the unlabeled documents from the last hidden layer of a [MASK] token in the template. In this way, category embeddings of both category names and unlabeled documents become contextualized token embeddings that can be mapped onto a unified category embedding space, on which classification can be easily conducted by similarity comparison. Here, resolving semantic gaps between the two types of embeddings through alignment, like X-Class [5] and SimPTC [9], is not necessary.

Nevertheless, the category embeddings predicted by the original PLM from the [MASK] token are not clearly separated in the embedding space, thus further refinement of the embeddings is required. To deal with this issue, we adopt contrastive learning, which is known to be effective for capturing similarity relationship [10, 11, 12], to separate predicted category embeddings into clusters. However, training the PLM with only traditional contrastive learning loss inevitably hurts its encoding ability that learned in the pretraining stage, leading to the collapse of the predefined category embeddings. Therefore, we propose a new contrastive learning loss function, named MLM-maintained contrastive loss, which incorporates a masked language modeling (MLM) loss term, aiming at maintaining the
encoding ability of the PLM, so that the quality of the predefined category embeddings is preserved.

There is another challenge when no annotated data is accessible, such that we have no validation set to check the classification ability of the model during the training. To resolve this problem, we newly propose confidence factor which is used as an indicator to estimate the intermediate classification performance. We record the confidence factor after every certain training steps during the training process, and retain the model having the highest confidence factor to predict category embeddings from unlabeled documents and category names. We name this process confidence-driven contrastive learning.

After the confidence-driven contrastive learning, each document can be assigned a pseudo label by searching the most semantically similar predefined category name. The pseudo labels are used to finetune the final classifier, but only confident pseudo labels are used and progressively added, to alleviate the influence of noisy pseudo labels involved in earlier stages.

The main contributions of this paper are summarized as:
1. We introduce a method to train a PLM to predict category embeddings from category names and unlabeled documents, based on which classification can be easily conducted by similarity comparison without the need for embedding alignment.
2. Confidence-driven contrastive learning is proposed to refine the category embeddings, in which MLM-maintained contrastive loss is introduced to improve category separation while preventing collapse of category embeddings, and confidence factor is introduced to estimate the classification ability of the model.
3. A progressive self-training strategy is introduced to alleviate the problem of amplifying noise during self-training.
4. The evaluation results on five representative datasets show our proposed Con2Class exceeds those of known models by 5.8% and 5.6% in average on macro and micro F1-scores, respectively.

The rest of this paper is organized as follows: Section 2 covers related work. Section 3 formulates the task and introduces our proposed method Con2Class. Section 4 presents the experiments and results. Section 5 shows analysis and discussion on Con2Class. Section 6 gives a conclusion and discusses future directions.

2. Related work

Pretrained language models (PLMs) are derived from transfer learning, which is originally proposed with the aim to solve a new problem using limited labeled samples [13]. Transformer-based PLMs, BERT [14] as an example, are pretrained by objectives such as next sentence prediction (NSP), masked language modeling (MLM), etc., to obtain linguistic knowledge from huge general corpora, such that PLMs can achieve remarkable results on specific tasks by finetuning on limited task-specific data [13, 15, 16].

NSP is designed to distinguish whether two sentences are arranged in the correct order, through the output of a [CLS] token. The [CLS] token contains the information of all the words in the sentence, which is usually used for sentence-level representation and classification.

Given a document \( \mathbf{d} \) in which several words are masked by [MASK] tokens, MLM aims to predict the likelihood of each word \( w \) appears at the [MASK] position over all the vocabulary:

\[
p(w | \mathbf{d}) = \text{softmax}(W_2 \sigma(W_1 \mathbf{d} + \mathbf{b})),
\]

where \( W_1, W_2 \) and \( \mathbf{b} \) are trainable parameters and \( \sigma(\cdot) \) is the sigmoid function. By virtue of MLM training objective, prompt learning, which leverages prompting texts-included templates to probe into the knowledge of PLMs, is prevailing for its easy deployment and lighter requirement for annotated samples [17]. Such a paradigm is in the limelight among wide fields like text classification [18, 19], summarization [20] and question answering [21]. Researches revolving around prompt improvement are also widely conducted [3, 22, 23, 24, 25].

Through the Transformer architecture [26], PLMs can capture high-order, long-range dependencies over texts and generate context-aware word representations, known as contextualized word embeddings, which are more informative than traditional word embeddings such as Word2Vec [27] or Glove [28]. Additionally, by leveraging techniques such as Siamese network [29], prompt learning [11] and contrastive learning [10, 12], PLMs can also generate informative sentence embeddings, which can be used for classification, natural language inference and semantic similarity tasks [10].

Contrastive learning. To better capture the semantic relationship between sentences, contrastive learning is widely adopted under the scenario of unsupervised learning. ConSERT [12] proposes four augmentation strategies including adversarial attack, token shuffling, cutoff, and dropout, to construct positive pairs and captures semantic relationship through the average pooling over BERT’s last hidden layer. SimCSE [10] employs the [CLS] token to learn semantic relationship between documents, in which standard dropout [24] is applied twice on one document to generate two different embeddings used as positive pairs. Embeddings produced by different documents are regarded as negative pairs in their contrastive learning. RTS [31] adopts similar strategy with SimCSE in contrastive learning, but positive pairs are constructed by two segmentations randomly segmented from one document. To avoid interference from the length of the document, ESImCSE [32] improves SimCSE by executing word repetition to differentiate the length of documents and retain the original meaning. Additionally, momentum contrast is proposed to expand negative pairs for more refined contrastive learning.
PromptBERT [11] utilizes text templates to learn semantic relationship through a [MASK] token. Templates are manually designed to fit for task objectives, and multiple embeddings are extracted from the same sentence for positive pairs.

**Classification using only category names.** The approach of nearest neighbor search is widely adopted in category-name-only classification. One of the most representative works is Prototypical Networks [33] in the field of computer vision, in which images are classified by comparing similarities with class prototypes. In text classification, analogously, documents are classified by comparing semantic similarities with category names, for which document and category representations are essential to the success of the classification.

Table 1 summaries the differences between existing category-name-only classification methods and our method Con2Class proposed in this paper, on the points such as embedding generation methods, alignment and contrastive learning, etc. Traditional encoding methods such as Bag-of-Words (BoW) [27], Explicit Semantic Analysis (ESA) [34], Glove [28] and Latent Semantic Analysis (LSA) [35] are adopted in early works such as Dataless [7] and the method introduced in [8], to encode documents and category names. However, these encoding methods can hardly capture the contextualities of semantic representations. Recently, PLMs becomes popular in text representation generation, for their strong ability on capturing contextual relationship. RTS [31] employs the [CLS] token of MPNet [36] to obtain the embeddings for documents and categories, but it works only when the description of categories is available since only one word or phrase can hardly provide enough semantic information for encoding. X-Class [5] constructs embeddings by the weighted average of contextualized token embeddings, but in this way, document embeddings are not necessarily placed around their category embeddings, for which an alignment process by Gaussian Mixture Model (GMM) [37] is necessary to be executed. SimPTC [9] fills category names into hand-crafted templates to construct sentences such that the embeddings of both documents and categories can be represented by the [CLS] token. GMM is also conducted for alignment in SimPTC.

Apart from the nearest neighbor search, there are also methods categorizing documents using word-level information by counting term frequency which is demonstrated by ConWea [6] and matching category-related words which is proposed in LOTClass [4]. However, concrete words are less informative than dense embedding vectors, resulting in higher level of misclassification.

3. Task definition of category-name-only classification and methodology

Suppose a collection of documents \( X = \{x_1, x_2, ..., x_n\} \) is assigned with a collection of category names \( C = \{c_1, c_2, ..., c_m\} \), by a mapping \( M : X \mapsto C \). The goal of category-name-only classification is to train a prediction model that predicts \( M \), but no training data that associate documents \( X \) with category names \( C \) are given.

![Fig. 1](image-url) The architecture of our proposed Con2Class.

In this paper, we propose a nearest neighbor search-based model Con2Class to solve this tough task. Fig. 1 depicts the conceptual architecture of Con2Class. We first finetune a pretrained language model (PLM) BERT-em through confidence-driven contrastive learning, to produce embeddings for all the predefined category names and predict category embeddings for all the unlabeled documents in a unified category embedding space. Then, pseudo labels are assigned by searching the closest predefined category embedding in this unified category embedding space, without the need of alignment. Finally, a progressive self-training strategy is introduced to train another PLM classifier BERT-cl for final prediction. Note that Con2Class is PLM-antagonistic and in this paper, we employ the representative PLM BERT [14], for fair comparison with other methods.

3.1 Category embedding production

As introduced in Section 2, nearest neighbor search-based methods usually compare similarities between documents and categories, which requires alignment to narrow the semantic gap between document embeddings and category embeddings. In this paper, instead, we produce predefined category embeddings and predicted category embeddings in a unified category embedding space, such that similarity comparison can be conducted without the need of alignment. Fig. 2 illustrates how predefined category embeddings and predicted category embeddings are produced when the

<table>
<thead>
<tr>
<th>Method</th>
<th>Document/Category embedding</th>
<th>Alignment</th>
<th>Template</th>
<th>Contrastive learning (Loss function)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataless</td>
<td>BoW/ESA</td>
<td>Not implemented</td>
<td>Not used</td>
<td>Not used</td>
</tr>
<tr>
<td>RTS</td>
<td>[CLS] token</td>
<td>Not implemented</td>
<td>Not used</td>
<td>Used ((\ell_{CC}))</td>
</tr>
<tr>
<td>X-Class</td>
<td>Weighted average on MLM prediction</td>
<td>Implemented</td>
<td>Not used</td>
<td>Not used</td>
</tr>
<tr>
<td>SimPTC</td>
<td>[CLS] token</td>
<td>Implemented</td>
<td>Used</td>
<td>Not used</td>
</tr>
<tr>
<td>Con2Class (ours)</td>
<td>MLM prediction</td>
<td>Not implemented</td>
<td>Used</td>
<td>Used (((1 - a)\ell_{CL} + a\ell_{MLM}))</td>
</tr>
</tbody>
</table>

Copyright © 20XX The Institute of Electronics, Information and Communication Engineers
template “[X] is about [MASK]” is used, in which [X] refers to the raw document.

### Predefined category embedding

RTS [31] regards category descriptions as documents and generates predefined category embeddings through the [CLS] token. However, it is impractical to obtain informative category embeddings from the [CLS] token under our setting that using only category names. SimPTC [9] fills category names into hand-crafted templates to construct anchor sentences with the aim of obtaining predefined category embeddings from the [CLS] token. But the semantics of these anchor sentences deviate to some extent from the original meanings of the category names, which is not desirable for nearest neighbor search-based classification. Therefore, we choose to employ a BERT model named BERT-em to produce contextualized category embeddings leveraging the unlabeled documents which contain at least one category name. Suppose the category name $c_j$ appears a total of $q_j$ times in the corpus. The embedding of $c_j$ is calculated by averaging the hidden states of the last hidden layer on all the occurrences of category name $c_j$ in the corpus:

$$e_{c_j} = \frac{1}{q_j} \sum_{p=1}^{q_j} h_{c_j}^p,$$  \hspace{1cm} (1)

where $h_{c_j}^p$ is the state of the last hidden layer of $c_j$ at the $p$-th appearance in corpus. It can be extended to the case that the category name is a phrase by averaging the hidden states on all the words in the phrase.

### Category embedding prediction

To facilitate similarity comparison with predefined category embeddings, we consider predicting category embeddings of unlabeled documents into a unified category embedding space. Inspired by PromptBERT [11], we manually design a prompt-contained template, in which a [MASK] token is included to predict the category embedding $e_i$ of document $x_i$ from the last hidden layer of the [MASK] token $h_{\text{[MASK]}}$:

$$e_i = h_{\text{[MASK]}},$$  \hspace{1cm} (2)

In this way, both predefined category embeddings and predicted category embeddings are contextualized token embeddings generated by MLM prediction, whose proximities can be measured without the need of alignment.

Note that in PromptBERT, using an identical model like BERT-em for generating category embeddings for both documents and category names is not considered.

### 3.2 Confidence-driven contrastive learning

#### Refining predicted category embeddings

The template “[X] is about [MASK]” in Fig. 2 guides the BERT-em model to predict category embeddings of unlabeled documents in the category embedding space. Nevertheless, the category embeddings predicted by the raw BERT-em model are not clearly separated by categories, which makes it difficult to distinguish category membership by nearest neighbor search. Therefore, we finetune the BERT-em model by contrastive learning, which has been proved to effectively capture the similarity relationship between documents [10, 11, 12], to separate predicted category embeddings in the category embedding space.

Fig. 3 shows the schema of our contrastive learning. As introduced in Section 2, there are many ways to construct positive and negative document pairs for contrastive learning. In our work, we follow SimCSE [10], an easy and effective method, to construct document pairs leveraging standard dropout [30]. Each document is fed into BERT-em twice and the two outputs generated with different dropout masks constitute a positive pair. The outputs of different documents within a mini-batch, that are fed into BERT-em at the same time, constitute negative pairs. The training target is to push apart the predicted category embeddings of different documents (negative pairs) and pull close the predicted category embeddings of the same document with different dropout masks (positive pairs).

Suppose $x_i$ and $x_j$ are documents in a mini-batch of size $B$, $h_i^z$ and $h_j^z$ are the predicted category embeddings of $x_i$ and $x_j$ with different dropout masks $z$ and $z'$, then the loss function of contrastive learning for one mini-batch of size $B$ is:

$$L_{CL} = \frac{1}{B} \sum_{i=1}^{B} \log \frac{e^{\frac{\text{sim}(h_i^z, h_j^z)}{\tau}}}{\sum_{j=1}^{B} e^{\frac{\text{sim}(h_i^z, h_j^z)}{\tau}}},$$  \hspace{1cm} (3)

where $\tau$ is a temperature hyperparameter.

However, roughly applying contrastive learning for separating predicted category embeddings inevitably hurts
the encoding ability learned from MLM pretraining objective, which can collapse predefined category embeddings and deteriorate nearest neighbor search.

For this reason, we improve the loss function by adding the original MLM loss of BERT to the contrastive learning loss, trying to maintain the encoding ability of the BERT-em during contrastive learning, thus, preventing the collapse of predefined category embeddings:

$$L = (1 - \alpha) \mathcal{L}_{CL} + \alpha \mathcal{L}_{MLM}.$$  \hspace{1cm} (4)

We name the loss function of Eq. (4) **MLM-maintained contrastive loss**, where $\alpha$ is a hyperparameter weighting two losses.

**Prediction confidence.** We train the BERT-em model to produce category embeddings from both predefined category names and unlabeled documents, on which a classifier based on nearest neighbor search is trained. However, there is no annotated data that can be used as a validation set to check the classification accuracy during training, for which we need to find a substitution for estimating the classification ability of the model. Intuitively, a good classifier should be able to clearly distinguish the category that a document belongs to. Here, we consider deriving a classification quality indicator from distributions of confidence scores, for which we newly propose **confidence factor (CF)** to estimate the classifier performance. For calculation of CF, we firstly evaluate semantic similarities between predicted category embeddings and predefined category embeddings. Cosine similarity is adopted to evaluate the similarity between the predicted category embedding of document $x_i$ and the predefined category embedding of category $c_j$:

$$s_i^j = \cos (e_i, e_j).$$  \hspace{1cm} (5)

The predicted probability distribution $p_i \in \mathbb{R}^m$ of $x_i$ is then calculated by z-score normalization and softmax function, applied on the similarity vector $s_i = [s_i^1, s_i^2, ..., s_i^m]$:

$$z(s_i) = \frac{s_i - \mu}{\sigma}.$$  \hspace{1cm} (6)

$$p_i = \text{softmax}(z(s_i)),$$  \hspace{1cm} (7)

where $\mu$ and $\sigma$ are the mean value and the standard deviation of $s_i$. Z-score normalization here aims to sharpen the probability distribution to magnify the belonging category more clearly.

We define document-wise confidence score ($CS$) to evaluate the confidence of a single prediction on document $x_i$:

$$CS_i = \max p_i.$$  \hspace{1cm} (8)

The confidence factor is then calculated by averaging the confidence scores of all the documents in $X$:

$$CF = \frac{1}{n} \sum_{i=1}^{n} CS_i.$$  \hspace{1cm} (9)

Note that $CF$ becomes constant in the case of binary classification ($m = 2$), losing its role, for which we skip z-score normalization and directly apply softmax function on the similarity vector $s_i$ as Eq. (10) presents to calculate the probability distribution for binary classification:

$$p_i = \text{softmax}(s_i).$$  \hspace{1cm} (10)

We train the BERT-em model for a total of $\Phi$ training steps using unlabeled documents iteratively, and evaluate $CF$ every $\varphi$ training steps, such that there are a total of $\lfloor \Phi/\varphi \rfloor$ checks throughout the training, where $\lfloor \cdot \rfloor$ is the floor function. Finally, the model having the highest $CF$ is used to produce predefined category embeddings and predicted category embeddings for label prediction. We name the above training process **confidence-driven contrastive learning**.

### 3.3 Progressive self-training by pseudo labels

**Progressive self-training** executes a cycle of classifier training and pseudo label generation, to progressively improve classification performance. To train a classifier, we employ a separate BERT model BERT-cl, on which a classifier head is added. We finetune the BERT-cl classifier following the instruction by [14]. At the initial stage of self-training, the BERT-cl classifier is finetuned under the supervision of pseudo labels generated by confidence-driven
contrastive learning. From the second self-training loop, the pseudo labels and the confidence scores are produced by the probability distribution predicted by the BERT-cl classifier itself. Since noisy pseudo labels that indicate incorrect categories may exist, we add a confidence regularizer $\mathcal{R}(\theta)$ weighted by $\lambda$ to the classical Kullback-Leibler divergence to finetune the BERT-cl classifier with the aim of avoiding overfitting on those noisy pseudo labels, as [38] did. Thus, the loss function for the BERT-cl classifier finetuning is:

$$L_{FT}(\theta; \tilde{y}) = \mathcal{D}_{KL}(\tilde{y} || f(x; \theta)) + \lambda \mathcal{R}(\theta),$$

$$\mathcal{R}(\theta) = \mathcal{D}_{KL}(u || f(x; \theta)).$$

(11)

where $\tilde{y}$ refers to the pseudo label, $u \in \mathbb{R}^m$ is uniform distribution in which $u_i = 1/m$, for $i = 1, 2, \ldots, m$.

Since the overall accuracy of pseudo labels is relatively low at the initial stage, we introduce a progressive training strategy such that top $(P + 10 \times t)$ percent of the predictions having the highest confidence scores are used for the BERT-cl classifier finetuning in the $t$-th self-training loop, where $P$ is a pre-defined hyperparameter. In this way, only a small part of predictions having high-confidence pseudo labels are leveraged at the beginning of the self-training, and the scale of pseudo labeled documents expands as the self-training proceeds. To avoid noise amplification from wrongly marked documents in previous loops, the BERT-cl classifier finetuning always starts again from the original pretrained BERT model at each self-training loop. The self-training will be terminated when it reaches the maximum count of iteration $T$.

4. Experiments

4.1 Datasets

We conduct experiments on five representative datasets: AGNews [39], 20News [40], Yelp [39], IMDB [41] and DBpedia [39], covering the fields of news, review and ontology. The features and statistics are listed in Table 2.

4.2 Baseline methods

We compare our proposed Con2Class with several advanced methods targeting at the same task of categorizing using only category names as supervision.

WESTCLASS [42] assumes that words and documents share a joint semantic space. Pseudo-labeled documents are generated from the semantic space and then, leveraged for classifier training under the self-training manner. ConWea [6] uses contextualized representations and seed word information to distinguish polysemes of a word and create a contextualized corpus, which is used to train the classifier and expand seed words iteratively. LOTClass [4] leverages pretrained language models to generate category vocabulary, and then, search category-indicative words through matching replaceable words with the words in the category vocabulary as training supervision. X-Class [5] generates class-oriented representation for documents and align documents to classes through clustering. SimPTC [9] embeds category names in templates to construct anchor sentences, and classification is implemented by clustering texts in embedding space. Following the setting of the original paper, we run SimPTC using RoBERTa-large model on our dataset. Additionally, we present the result of supervised learning, in which a BERT classifier is finetuned by a 2-fold cross-validation setting.

4.3 Experimental settings

Since the domains of the benchmark corpora are diverse, we manually design different templates to adapt to different types of corpora. As shown in Table 3, the classification task on five datasets is roughly divided into two types: topic classification (AGNews, 20News and DBpedia) and sentiment analysis (Yelp and IMDB). With PET [18] and PromptBERT [11] as references, two templates are designed for each subtask in our experiments.

<table>
<thead>
<tr>
<th>Task type</th>
<th>Dataset</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic classification</td>
<td>AGNews, 20News, DBpedia</td>
<td>The topic of [X] is about [MASK].</td>
</tr>
<tr>
<td>Sentiment analysis</td>
<td>Yelp, IMDB</td>
<td>[X] All in all, it was [MASK].</td>
</tr>
</tbody>
</table>

All the experiments are conducted using one NVIDIA GeForce RTX 3090 GPU. For both BERT-em and BERT-cl, the BERT model of ‘bert-base-uncased’ is adopted. The hyperparameter $\alpha$ in the contrastive learning loss is set to 0.1 in all the experiments, because BERT-em has already been trained by the MLM objective in the pretraining stage and we just need to maintain the ability of MLM prediction in confidence-driven contrastive learning. The temperature hyperparameter $\tau$ is set to 1 in all the experiments. On account of the limitation of GPU memory, we set mini-batch size $B = 10$ in contrastive learning and update BERT-em parameters every 10 mini-batches, making up one training step. Following this setting, the batch size of confidence-driven contrastive learning is practically 100. Since the DBpedia dataset has much more categories than the other datasets, we conduct confidence-driven contrastive learning a total of $\Phi = 100,000$ training steps for the DBpedia dataset and $\Phi = 10,000$ training steps for the other datasets. The confidence factor is evaluated every


\[ \varphi = 10,000 \] training steps for the DBpedia dataset and \[ \varphi = 1,000 \] training steps for the other datasets, such that the confidence factor is evaluated a total of 10 times throughout the confidence-driven contrastive learning for all the datasets. The batch size of the BERT-cl classifier finetuning is set to 24 and BERT-cl parameters are updated every 5 batches. The learning rates of the contrastive learning and the BERT-cl classifier finetuning are 3e-5 and 1e-5, respectively. Since larger datasets may incur more noises in pseudo-labeled data, we set \[ P = 20 \] for DBpedia for its large dataset size, while \[ P = 40 \] for the other datasets. The weighting hyperparameter \( \lambda \) is set to 0.1 and the BERT-cl classifier is finetuned 5 epochs for DBpedia, while 10 epochs for the other datasets, in consideration of the dataset sizes as well. Progressive self-training is conducted for \( T = 3 \) iterations in our experiments.

Since further pretraining helps improve the classification performance [15], we further pretrain the BERT-cl model 10 epochs on the DBpedia dataset and 100 epochs on the other datasets before the BERT-cl classifier finetuning. Since there is a huge gap in styles between human reviews and the original pretraining data of BookCorpus [43] and Wikipedia, we also further pretrain the BERT-em model before confidence-driven contrastive learning on the Yelp and IMDB datasets. For all the datasets, we execute five runs and compare on the average scores.

### 4.4 Experimental results

We compare our proposed Con2Class with recent advanced methods by macro-F1 and micro-F1 scores, among which LOTClass [4], X-Class [5], and SimPTC [9] share the SOTA records over all the datasets. Apart from the full version of Con2Class, we additionally evaluate the performance of confidence-driven contrastive learning alone, in which the progressive self-training is removed from Con2Class, to demonstrate the classification ability by conducting only confidence-driven contrastive learning.

According to the results in Table 4, Con2Class surpasses all the baselines over the five datasets. The results on the Yelp and IMDB datasets are even approaching that of supervised BERT finetuning, demonstrating the strong ability of Con2Class on the task of classification using category names only.

Notably, the results of confidence-driven contrastive learning only, on the Yelp dataset, have already exceeded that of baselines, which is further improved by the succeeding progressive self-training. This could be attributed to the clear and distinct meaning of their category names “Good” and “Bad”.

In the ablation study, we remove the MLM loss function \( \mathcal{L}_{\text{MLM}} \) from Eq. (4) for both confidence-driven contrastive learning and Con2Class. According to the results, the classification ability of confidence-driven contrastive learning is degraded after \( \mathcal{L}_{\text{MLM}} \) is removed, which further affects the progressive self-training, reflected in the performance drop on Con2Class. Interestingly, the result of the confidence-driven contrastive learning of the 20News dataset is almost unchanged after \( \mathcal{L}_{\text{MLM}} \) is removed, but the performance of Con2Class drops remarkably. It reflects the fact that the effect of progressive self-training is not entirely governed by the accuracy on the whole samples, but it is directly related to the accuracy of those top-ranked high-confidence pseudo-labeled samples.

The time cost of confidence-driven contrastive learning is affected by pre-set hyperparameters such as the training step \( \Phi \) and the interval step \( \varphi \) between two confidence score checks. For the cases when \( \Phi = 10,000 \) and \( \varphi = 1,000 \) on the AGNews, 20News, IMDB and Yelp datasets, confidence-driven contrastive learning takes about 3 hours. For the case when \( \Phi = 100,000 \) and \( \varphi = 10,000 \) on the DBpedia dataset, the time consumption increases with the expansion of training steps, reaching 30 hours. The time cost of progressive self-training varies with the scale of the corpus and it ranges from 1 hour on the smallest dataset 20News to 12 hours on the largest dataset DBpedia.

### 5. Discussion and analysis

#### 5.1 Insight of confidence-driven contrastive learning

To explore how the classification ability changes during the confidence-driven contrastive learning, we evaluate macro-F1, micro-F1 and the confidence factor after every training step on the 20News dataset.

The evaluation results on the first 100 training steps are

<table>
<thead>
<tr>
<th>Model</th>
<th>AGNews</th>
<th>20News</th>
<th>Yelp</th>
<th>IMDB</th>
<th>DBpedia</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised BERT finetuning</td>
<td>93.98/93.99</td>
<td>96.45/96.42</td>
<td>95.09/95.80</td>
<td>92.27/92.28</td>
<td>98.96/98.96</td>
</tr>
<tr>
<td>WeSTClass [42]</td>
<td>82.1*/82.3*</td>
<td>69.90*/71.28*</td>
<td>81.67*/81.6*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ConWea [6]</td>
<td>74.2*/74.6*</td>
<td>73.26*/75.73*</td>
<td>71.27*/71.4*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LOTClass [4]</td>
<td>86.82*/86.89*</td>
<td>72.53*/73.78*</td>
<td>87.68*/87.75*</td>
<td>85.97/85.98</td>
<td>85.98*/86.66*</td>
</tr>
<tr>
<td>X-Class [5]</td>
<td>85.66*/85.74*</td>
<td>77.76*/78.62*</td>
<td>90.09*/90.0*</td>
<td>82.93/82.95</td>
<td>91.17*/91.32*</td>
</tr>
<tr>
<td>SimPTC [9]</td>
<td>87.25*/87.32</td>
<td>82.84/84.87</td>
<td>94.29/94.29</td>
<td>90.90/90.90</td>
<td>91.88/92.14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Baselines</th>
<th>Confidence-driven contrastive learning</th>
<th>AGNews</th>
<th>20News</th>
<th>Yelp</th>
<th>IMDB</th>
<th>DBpedia</th>
</tr>
</thead>
<tbody>
<tr>
<td>72.53*/73.78*</td>
<td>82.48/82.72</td>
<td>75.13/76.17</td>
<td>94.48/94.48</td>
<td>85.49/85.64</td>
<td>84.03/84.81</td>
<td></td>
</tr>
<tr>
<td>69.90*/71.28*</td>
<td>87.29/87.32</td>
<td>82.87/83.60</td>
<td>95.76/95.76</td>
<td>91.25/91.27</td>
<td>94.07/94.15</td>
<td></td>
</tr>
</tbody>
</table>

#### Results marked with * are from the paper [5]
shown in Fig. 4. The F1 scores are at a low level when no training is done. Then they increase rapidly through the first few training steps and fluctuate around a relatively high level after 20 training steps. The confidence factor slightly drops at first and then, increases along with F1 scores. After 20 training steps, it fluctuates and exhibits a similar variation trend with F1 scores, validating the effectiveness of using our proposed confidence factor to estimate the classification ability.

Fig. 4 The changes in F1 scores and confidence factor.

There is a noticeable drop in the confidence factor at the beginning of the training. It can be attributed to the fact that the initial predicted category embeddings are not discernable enough, so that a great number of predicted category embeddings are unexpectedly close to a wrong predefined category in the embedding space, which causes low F1 scores. With the training proceeds, predicted category embeddings move away from wrong predefined categories and approach towards correct ones. Such a process causes a slight decrease on the confidence factor at the early stage as predicted category embeddings leave from wrong categories, after which, a subsequent boost on the confidence factor occurs as predicted category embeddings move towards correct categories.

5.2 Sensitivity analysis

Hyperparameter \( \alpha \). The loss function of the confidence-driven contrastive learning involves the hyperparameter \( \alpha \) weighting the contrastive learning loss \( L_{CL} \) and MLM loss \( L_{MLM} \). We change \( \alpha \) ranging on \([0.1, 0.3, 0.5, 0.7, 0.9]\) to explore how it affects the classification performance. Tests are conducted on all the five datasets and the results are shown in Fig. 5.

Fig. 5 Macro-F1 and micro-F1 scores with varying \( \alpha \).

According to the results, the classification performance does not show remarkable differences with varying values of \( \alpha \) on the Yelp, IMDB and 20News datasets. On the AGNews dataset, however, the performance is not much changed until \( \alpha \) reaches 0.9, an obvious performance drop happens. Regarding the DBpedia dataset, F1 scores constantly decrease as \( \alpha \) rises.

Considering the result of the ablation study that removing \( L_{MLM} \) leads to performance drop, a small \( \alpha \) is preferable for our proposed Con2Class.

Manually designed templates. As introduced in Section 4.3, we design two templates for the two subtasks, respectively, to predict category embeddings. In this part, we investigate how templates affect the classification results.

Table 5 lists all the templates we used for investigation, which are designed with reference to PET [18] and PromptBERT [11]. T1 refers to the template adopted in our experiment, which is designed with consideration of the style of the corresponding subtask. T2-T4 are the other three templates designed based on the style of the corresponding subtask. Additionally, to explore the case that the template is not much compatible with the corpus, we exchange the T1 templates for the two subtasks, marked as T5.

Table 5 The templates designed for two subtasks.

<table>
<thead>
<tr>
<th>Topic classification task</th>
<th>Sentiment analysis task</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGNews, 20News, DBpedia</td>
<td>Yelp, IMDB</td>
</tr>
<tr>
<td>T1 The topic of ‘[X]’ is about [MASK].</td>
<td>[X] All in all, it was [MASK].</td>
</tr>
<tr>
<td>T2 The category of ‘[X]’ is about [MASK].</td>
<td>[X] All in all, it is [MASK].</td>
</tr>
<tr>
<td>T3 The type of ‘[X]’ is about [MASK].</td>
<td>[X] In summary, it was [MASK].</td>
</tr>
<tr>
<td>T4 ‘[X]’ is about [MASK].</td>
<td>[X] In summary, it is [MASK].</td>
</tr>
<tr>
<td>T5 [X] All in all, it was [MASK].</td>
<td>The topic of ‘[X]’ is about [MASK].</td>
</tr>
</tbody>
</table>

Fig. 6 Macro-F1 and micro-F1 scores using different templates.

Fig. 6 depicts the results of using different templates in our model. T1-T4, which we carefully designed with respect to the type of the corpus, are not showing remarkable differences on the classification performance. On the other hand, an obvious performance drop is observed on the first four datasets when the template is swapped with the other subtask, represented by T5. The result is as we envisioned because MLM prediction greatly relies on context and with
an irrelevant template, BERT-em can hardly produce embeddings correctly representing categories of documents.

The DBpedia dataset is not much influenced by the template T5. It can be attributed to the fact that each document in the DBpedia dataset is a paragraph describing an ontology, such that the fifth template “[X] All in all, it was [MASK].” can also work on this type of content.

5.3 Embedding space visualization

The performance of nearest neighbor classification is greatly influenced by the document/category distribution over the embedding space. Intuitively, it is desirable that documents cluster by category compactly and distinguish from other categories clearly.

In this part, we use t-SNE [44] to visualize the embedding space over five different methods: average pooling, X-Class, SimPTC, Con2Class and Con2Class without MLM loss. Fig. 7 shows the results on the 20News and DBpedia datasets. For each dataset, we randomly sample 10,000 documents for visualization and different categories are distinguished by different colors. Each dot point refers to one document and each diamond point refers to one category. For the averaging pooling method, documents are represented by the average of the BERT output layer and each category is represented by the contextualized embedding encoded by BERT. X-Class produces document and category embeddings by weighted average of contextualized embeddings and SimPTC obtains document and category embeddings from the BERT’s last hidden layer of the [CLS] token. For the methods of Con2Class and Con2Class without MLM loss, the document is represented by the predicted category embedding and the category is represented by the contextualized category embedding.

According to the visualization results, the boundary of document representations produced by the averaging pooling is fuzzy and documents of different categories are not clearly separated. Moreover, a great part of category representations appear at the boundary of two categories, which is unfavorable for category assignment. The following three methods show much more clear boundaries between different categories and most of the documents gather around the correct categories. Benefiting from contrastive learning, Con2Class is able to produce even more compact clusters, which signifies the category separation of documents, such that wrongly assigned pseudo-labels on wavering documents are less involved in the succeeding BERT-cl finetuning. Without the MLM loss, however, the boundaries between different categories of documents become blurred and a larger proportion of categories appear at boundaries or wrongly gather with irrelevant documents, which proves that MLM loss is helping separate categories of documents and prevent the collapse of category embeddings.

6. Conclusion

To tackle the task of document classification using category names only, a new method named Con2Class is proposed in this paper. For nearest neighbor search, a pretrained language model is finetuned through confidence-driven contrastive learning to produce predefined category embeddings and predict category embeddings of documents in a unified embedding space with the help of prompt-contained templates. MLM-maintained contrastive loss is introduced in confidence-driven contrastive learning to refine predicted category embeddings and prevent the collapse of the predefined category embeddings during contrastive learning. The confidence factor is newly proposed to estimate the classification ability of the model, such that the model producing the most certain prediction, which we regard having the best classification ability, is selected to produce category embeddings for similarity evaluation and comparison. Pseudo labels obtained by...
searching the semantically closest predefined category name are used for classifier finetuning, based on which a progressive self-training is executed where high-confidence predictions are progressively added for classifier finetuning, while reducing the interference from noisy pseudo labels.

The performance evaluations show that Con2Class surpasses the SOTA results on all the five representative datasets. Intermediate results reveal the feasibility of using the newly proposed confidence factor to estimate the classification ability of the model. The sensitivity analysis reveals the impacts of the hyperparameter $\alpha$ and templates on the result, which demonstrates that a small $\alpha$ is preferable for Con2Class and an appropriate template is necessary to the success of Con2Class. Visualization on embedding space reveals that our proposed Con2Class is improving category separation for the unlabeled documents over the existing methods. Adding MLM loss to contrastive learning is also proved to effectively prevent the collapse of predefined category embeddings.

Nevertheless, these is still wide space to improve the results. For example, techniques of expanding the category names and designing multiple templates for ensemble, which have been proved to be effective in other works [4, 7, 8, 9], are supposed to boost the classification performance. Checking the confidence factor more frequently is also supposed to have a greater likelihood of finding superior models, yielding better results. Besides, replacing the concrete templates with more flexible continuous templates may contribute to a significant improvement on the task of classification using category names only, which is the direction we plan to move forward in our future work.

Acknowledgments

This work is in part supported by JSPS KAKENHI Number JP22K12044.

References


DOI:10.1145/3269206.3271737


Zhewei Xu received his B.S. degree in Computer Science and Technology from Harbin Engineering University and his M.S. degree in Energy and Environmental Engineering from Zhejiang University, in 2015 and 2018 respectively. He is currently a Ph.D. student in the Graduate School of Information, Production and Systems, Waseda University. His research interests include text classification, few-shot learning, and semi-supervised learning.

Iwaihara Mizuho received his B.Eng., M.Eng. and D.Eng. degrees all from Kyushu University, in 1988, 1990, and 1993 respectively. He was a research associate and then an associate professor in Kyushu University, from 1993 to 2001. From 2001 to 2009, he was an associate professor at Department of Social Informatics, Kyoto University. Since 2009, he is a professor at Graduate School of IPS, Waseda University. He is a member of IEICE, IPSJ, ACM and IEEE CS.

List of captions for all figures and tables

Fig. 1 The architecture of our proposed Con2Class.
Fig. 2 Extracting predefined and predicted category embeddings.
Fig. 3 The schema of contrastive learning.
Fig. 4 The changes in F1 scores and confidence factor.
Fig. 5 Macro-F1 and micro-F1 scores with varying α.
Fig. 6 Macro-F1 and micro-F1 scores using different templates.
Fig. 7 Embedding space visualization over five different methods.

Table 1 Comparison between existing category-name-only classification methods.
Table 2 Statistics of benchmark datasets.
Table 3 Manually designed templates where [X] is the raw text.
Table 4 Macro-/Micro-F1 scores on baselines, our methods, and ablation studies.
Table 5 The templates designed for two subtasks.