EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

Sentence classification based on the concept kernel attention mechanism

Hui Li[1], Guimin Huang[1,*], Yiqun Li[1], Xiaowei Zhang[1] and Yabing Wang[1]

[1]Guangxi Key Laboratory of Image and Graphic Intelligent Processing, School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China

Abstract

Sentence classification is important for data mining and information security. Recently, researchers have paid increasing attention to applying conceptual knowledge to assist in sentence classification. Most existing approaches enhance classification by finding word-related concepts in external knowledge bases and incorporating them into sentence representations. However, this approach assumes that all concepts are equally important, which is not helpful for distinguishing the categories of the sentence. In addition, this approach may also introduce noisy concepts, resulting in lower classification performance. To measure the importance of the concepts for the text, we propose the Concept Kernel Attention Network (CKAN). It not only introduces concept information into the deep neural network but also contains two attention mechanisms to assign weights to concepts. The attention mechanisms are the text-to-concept attention mechanism (TCAM) and the entity-to-concept attention mechanism (ECAM). These attention mechanisms limit the importance of noisy concepts as well as contextually irrelevant concepts and assign more weights to concepts that are important for classification. Meanwhile, we combine the relevance of concepts and entities to encode multi-word concepts to reduce the impact of the inaccurate representation of multi-word concepts for classification. We tested our model on five public text classification datasets. Comparison experiments with strong baselines and ablation experiments demonstrate the effectiveness of CKAN.

Keywords: Sentence classification, text conceptualization, concept knowledge base, attention mechanism, concept embeddings

Received on 21 March 2022, accepted on 08 May 2022, published on 17 May 2022

doi: 10.4108/eai.17-5-2022.173980

* Corresponding author. Email: sendhuang@126.com

1. Introduction

The widespread use of the Internet and mobile terminals has generated a huge amount of textual information. Among these texts, sentence text has become the main carrier for users to transmit information. These large amounts of sentence text contain a wealth of potentially valuable information. Correctly classifying sentence text can help uncover the potential value hidden in big data and help monitor online public opinion and information security [1-7].

Unlike documents or paragraphs, sentence text has limited contexts and lacks sufficient information for statistical inference. Recently, many works have combined conceptual knowledge from external knowledge bases to enrich the semantics of sentence text [8-13]. Concepts as high-level semantics can summarize entities with similar categories using concise meanings (e.g., Beyonce, Lady Gaga, R. Kelly all belong to a common concept *singer*). At the same time, different concepts can be assigned to ambiguous entities to distinguish different meanings (e.g., for the same entity, apple, there can be two different meanings, *fruit* and *company*, in the knowledge base). Wang et al. [10] proposed "Bag-of-Concepts", which constructed a concept space by mapping entities in a sentence to concepts in a taxonomy, and obtained a concept space representation of the sentence. Li et al. [14] proposed

automatically acquiring useful conceptual knowledge from Probase [15], conceptualizing words and phrases into concepts in a probabilistic manner, and eventually representing the sentence as a distributed vector in the learned concept space. Wang et al. [11] addressed the lack of is-A information in the sentence representation by combining explicit concept with an implicit sentence representation. Although concept-based sentence classification methods have made great progress, we argue that some problems have been overlooked in this stage of work.

First, the existing concept-based sentence classification methods do not consider the introduction of irrelevant or noisy concepts to the sentence. For example, given the sentence "Apple Shares 'Life is But a Dream' Shot on iPhone 13 Pro Film", we can find two different concepts of *fruit* and *company* for the entity "apple" in the knowledge base. Introducing the concept *fruit* into the sentence representation is not beneficial for the model to classify the sentence. Therefore, we should restrict the concepts that are irrelevant to the sentence and give them a lower weight in the representation of the concept set. Second, each entity in a sentence has different importance for determining the category of the sentence. Thus, the concepts corresponding to different entities should also be given different weights. For example, given the sentence "Extreme Jeep Wrangler prototype caught testing in Michigan," is found in the MIND dataset [16]. We can obtain two entities "Jeep Wrangler" and "Michigan" by means of entity linking. Obviously, "Jeep Wrangler" is more useful than "Michigan" for classifying sentences into the category, "autos". Accordingly, the concept *small SUVs*, *mini SUVs* corresponding to "Jeep Wrangler" should be given higher weights in the concept set {*small SUVs*, *mini SUVs*, *state*, *northern state*}.

In this paper, we propose a **C**oncept **K**ernel **A**ttention **N**etwork (CKAN) for incorporating concept information into a sentence representation and employ the attention mechanisms to assign weights to concepts. In particular, we introduce the text-to-concept attention mechanism (TCAM) to measure the similarity of a sentence to a concept and eliminate concepts that are not relevant. Additionally, we design an entity-to-concept attention mechanism (ECAM) that assigns more weights to concepts corresponding to entities that are more important for the classification. Then, we design a soft switch to dynamically adjust both weights to generate the final weight for each concept.

Our research focus is to reduce the impact of noisy concepts and context-irrelevant concepts in the knowledge base on sentence classification by assigning weights to concepts. In addition, we observed that there are many multi-word concepts in the knowledge base. For example, *small SUVs*, *car brands*, *northern climate*, etc. If we use Word2vec [17] or GloVe [18], the out of vocabulary (OOV) problem will arise. Traditional solutions to this problem are to initialize the concept randomly [12], using charCNN [19, 20], or using the sub-word method [21, 22, 23]. However, the concept vectors generated by random initialization do not have semantic information. CharCNN exploits only character-level information, but not the semantic relationships between words. Sub-word method cannot handle the whole word, and it is difficult to learn the real semantics with insufficient training data. In order to represent multi-word concepts more precisely, we generate multi-word concept representations by combining the relationship between concepts and instances.

The model proposed in this paper is divided into three parts. First is a text encoder, which uses Sentence-BERT (SBERT) [24] to extract text features, and then an LSTM is used to encode the sentence semantics. Second is the concept extraction part, where we extract the entities in the sentence and then find the concepts corresponding to the entities in the knowledge base and encode them as vectors. Meanwhile, we combine the relationship between concepts and instances to generate multi-word concept representations for multi-word concepts. The next part is the concept encoding part, which is the most critical part of the model. We design two attention mechanisms to calculate the weights of each concept vector separately, and a soft switch dynamically adjusts the ratio of the two weights to obtain an optimal weight for each concept vector. Finally, we classify the sentence based on the sentence representation and its concepts.

The main works of this paper are summarized as follows:

1) We enrich the text representation with conceptual knowledge to assist in sentence classification. In particular, we introduce two attention mechanisms (TCAM and ECAM) to assign weights to concepts. We also set a soft switch to dynamically combine the two weights and obtain an optimal weight.
2) We design a concept representation method by combining concept and instance relevance to address the problem of inaccurate semantic representation of multi-word concepts.
3) We construct expensive experiments on five public datasets. Comparison experiments with strong baselines and ablation experiments demonstrate the effectiveness of CKAN.

The rest of this paper is organized as follows: Section 2 introduces related works. Section 3 introduces our approach. Section 4 describes the datasets and the experimental results. Conclusion and future work are presented in Section 5.

2. Related Works

2.1 Sentence Classification

Sentence-level text classification is a critical task for data mining and information security [1-7]. Ge et al. [6] and Yin et al. [4] conducted research on sentence-level text classification for database privacy protection and network security. In data mining, Zhang et al. [5] researched the robustness of sentence-level text classifiers. Furthermore,

they demonstrated that random forests can be even more vulnerable than SVMs, either single or ensemble. Due to the limited length of sentence text, traditional text classification methods are difficult to extract sentence features. The existing sentence text classification methods are mainly divided into two categories. One is based on topic modeling algorithms. The other method is based on deep learning algorithms.

The topic modeling-based sentence classification method extracts sentence topics using a topic model and then uses the extracted topic information to classify sentences. Li et al. [25] proposed LTM which can drive an adaptive aggregation process of sentence texts and simultaneously estimates other latent variables of interest. Rashid et al. [26] proposed a fuzzy topic modeling method based on fuzzy perspective for sentence-level classification. Gao et al. [27] designed a novel model called CRFTM for sentence text topic modeling. CRFTM not only develops a generalized solution to alleviate the sparsity problem by aggregating sentence text into pseudo-documents, but also leverages a CRF regularized model that encourages semantically related words to share the same topic assignment. Gao et al. [28] proposed a weighted Conditional random field regularized Correlated Topic Model(CCTM) for mining the topic information of sentence text.

Recently, sentence classification methods based on deep learning have been widely studied. Researchers capture different types of features by building complex neural network structures, and make full use of distributed representations and their limited contextual information. Zhou et al. [29] combined Bi-LSTM with a two-dimensional CNN network for capturing both the time-step dimension features and the vector-dimension features at the same time. Peng et al. [30] propose a novel attention mechanism that can filter sentence text noise effectively. Devlin et al. [21] proposed BERT which consists of a multilayer bidirectional transformer structure. BERT achieves SOTA performance in many natural language understanding tasks. Reimers et al. [30] found that the sentence vector representation of the sentence obtained by directly inputting the sentence into the BERT model did not have semantic features. They used Siamese and triplet network structures to derive semantically meaningful sentence embeddings.

Although BERT-based pre-trained language models can capture deep semantic information of the text, they are not strong enough to handle the ambiguity of the sentence text because of the limited contextual information. Moreover, they cannot handle new and rare words, as well as nonstandard terms (abbreviations, aliases, acronyms, etc.) in the absence of context. To address the above issues, researchers have introduced external knowledge into sentence representation to extend sentence features [14-21, 31]. Among them, using conceptual knowledge for sentence-level text classification has gained increasing attention. Wang et al. [32] proposed "bag-of-concepts" using concepts from the knowledge base to represent sentence text, and then used them as features for text classification. To incorporate concepts into implicit representation (distributed representation of text). Xu et al. [8] and Chen et al. [12] used a CNN and an LSTM, respectively, to incorporate contextually relevant external knowledge into text representation to aid sentence classification. Wang et al. [11] used a character-level CNN and introduced character information into a two-layer network to capture both explicit and implicit information. Although the above approach introduced concept information into the sentence representation, it ignored the effect of introducing concepts from the knowledge base that were not relevant to the sentence on the model classification. Moreover, it did not consider the difference in the importance of concepts corresponding to different entities for sentence classification. In our work, we design two attention mechanisms to measure the importance of concepts to better assist sentence classification.

2.2 concept embedding

There are many concepts consisting of multiple words in the concept knowledge base. If these concepts are directly sliced into individual words, and then word embeddings averaging is used, it makes the generated concept embedding semantically inaccurate. Additionally, since the concepts extracted in the knowledge base are discrete, there is a lack of context for semantic derivation of concept embedding. Chen et al. [12] used random initialization to generate concept embedding. However, random initialization cannot generate accurate concept embeddings. Wang et al. [11] and Li et al. [19] used the character embedding approach for concept embedding. However, there was a data sparsity problem. Additionally, in the case of small training samples, it is difficult for character embedding to learn effective concept representations. Some researchers used the sub-word [22, 33] to deal with the OOV problem. While, sub-words cannot handle the whole word, and it is difficult to learn the actual semantics for insufficient training data. Xu et al. [8] generated concept embedding by using the average of instance embedding. This approach could learn the semantics of the concept using the resources in the concept knowledge base. However, it ignored the difference between concept and instance representations. In this work, we designed a concept embedding method based on the concept-instance relationship. Compared with the existing concept embedding methods, our approach not only makes full use of the information in the knowledge base to generate concept embeddings but also introduces the differences between concept and instance vectors in the concept representation to better capture the semantics of the concepts.

3. The Concept Kernel Attention Network

The overall structure of our model is illustrated in Fig. 1. The sentence text is encoded by SBERT and fed into an LSTM to obtain the text representation. Meanwhile, the

entities can be extracted from the input sentence after entity recognition. We can extract entity-relevant concepts in the knowledge base to form a concept set. Then concept embeddings can be obtained through concept encoding. After giving weights to the concepts by two attention mechanisms, we concatenate each concept embedding to obtain the concept representation. Then, the text representation and the concept representation are concatenated and sent to a fully connected layer for classification through a residual network.

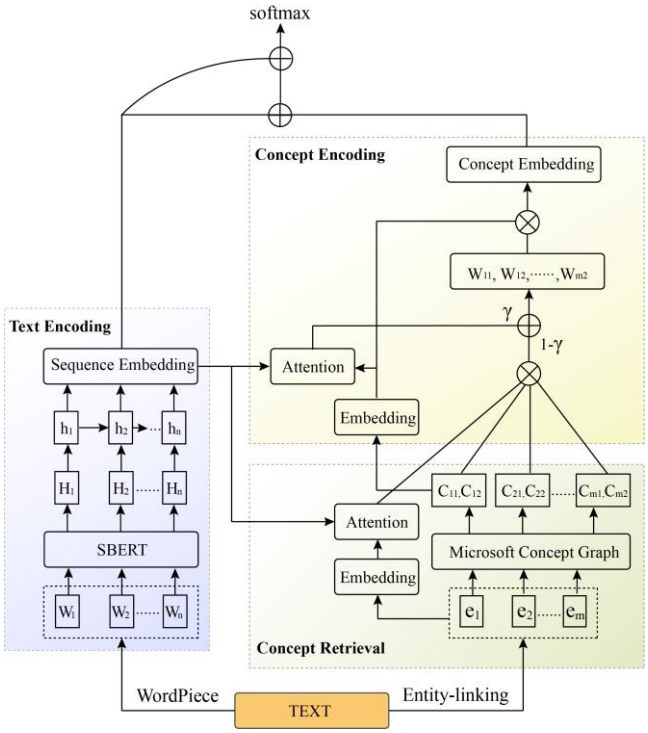


Figure 1. The structure of CKAN.

3.1. Concept retrieval

The aim of the concept retrieval module is to retrieve concepts for entities from a concept knowledge base. We adopt the Microsoft Concept Graph [34] as our concept knowledge base. The Microsoft Concept Graph is a large-scale probabilistic English concept knowledge base proposed by Microsoft Research Asia. It contains over 5 million concepts, 12 million entities and more than 87 million is-A relationships.

One important feature of the Microsoft Concept Graph is that concepts and entities are related in the way of probability. The probability score between an entity and a concept is represented by a typicality score, containing the probability $P(c|e)$ of the concept for a given entity and the probability $P(e|c)$ of an entity for a given concept. For example, $P(fruit|apple) > P(movie|apple)$ and $P(swallow|bird) > P(penguin|bird)$. Formally, typicality scores are derived from the frequency of co-occurrence between concepts and entities as follows:

$$P(c|e) = \frac{n(e,c)}{\sum_{e \in c_i} n(e,c_i)}, \quad (1)$$

$$P(e|c) = \frac{n(e,c)}{\sum_{e_i \in c} n(e_i,c)}. \quad (2)$$

Where $n(e,c)$ represents the frequency of co-occurrences between entity e and concept c in the web document.

The typicality score makes the knowledge representation more accurate and makes the query operation more flexible. However, when conceptualizing entities, the two typicality scores tend to give high scores to "extreme" concepts, i.e., generic or specific concepts. Given an entity e and $P(c|e)$ is proportional to $n(e,c)$, it tends to map e to generic concepts. $P(e|c)$ tends to give specific concepts that only contain e. However, generic concepts are less distinguishable, and specific concepts have fewer entities, which are not conducive to sentence classification. To find the "basic level concept", we conceptualize an entity using the improved conceptualization method proposed by Wang et al. [35], which is shown in Formula (3) as follows:

$$Rep(e,c) = P(c|e) \cdot P(e|c)_{k-smooth}. \quad (3)$$

Where $P(e|c)k\text{-}smooth$ is a smoothed typicality score, which can avoid P(e|c)-extracted special concepts covering very few entities. Formula (4) is written as follows:

$$P(e|c)_{k-smooth} = \frac{n(e,c)+k}{\sum_{e_i \in c} n(e_i,c) + kN_e}. \quad (4)$$

Where Ne is the number of all entities and k is a very small constant used to assume that each concept-entity pair has a small co-occurrence regardless of whether it is observed.

Given a sentence, we use stanza [36] to extract the entities in the sentence. Stanza is completely based on the neural network pipeline. The researchers pretrained it on 112 datasets, allowing stanza to achieve state-of-the-art results in several entity recognition tasks. For the extracted entities, we take out the top 5 highest scoring concepts based on $Rep(e,c)$.

3.2. Text encoding

Since we intend to use SBERT as the encoder, the format of the text input must also conform to it. We use WordPiece embeddings with a 30,000 token vocabulary to segment the input sequence. The input representation consists of three embedding layers: the token embedding layer, the segment embedding layer and the position embedding layer [21]. We suppose the input embedding layer is expressed as E_I, the token embedding is expressed as E_T, the segment embeddings are expressed as E_S, and the position embeddings are expressed as E_P; then, the corresponding formula is given as follows:

$$E_I = E_T + E_S + E_P. \quad (5)$$

SBERT extends the pretrained BERT model to obtain accurate sentence representations. In this paper, we use Sentence-BERT-base (SBERT-base) as the encoder. It consists of 12 transformer blocks and 12 self-attention heads. We initialize the component with the parameter of SBERT-base. The size of this parameter is 110 M. The input sequences are sent to SBERT to acquire a time-step sequence of hidden state vectors. Then, we fill the input layer of an LSTM with hidden state vectors to obtain the sentence vector representation $s \in \check{} \, ^{d_1}$.

3.3. Concept Encoding

In the Microsoft Concept Graph, due to its extensive coverage of concepts and instance pairs, the concepts and instances are often in a "one-to-many" relationship. For example, we can find multiple instances of the concept, *famous singer*, which are "celine dion", "britney spears", "anna vissi", etc. Concepts and instances are related by probability, which can help us generate concept embeddings. Here, we can represent a concept vector V_c as follows:

$$V_c=\{ e_1:w_1, e_2:w_2, ..., e_k:w_k \}$$

Where e_1, e_2..., e_k are the top k instances associated with the current concept that have been removed according to the typicality score $P(e|c)$. w_1,w_2...,w_k represent the relationship weights $P(e|c)$ between the instances and the concepts. For example, we can match the concept *famous singer* as a concept vector { celine dion : 0.0164 , britney spears : 0.0143 , anna vissi : 0.0123,…, johnny jordaan : 0.0020 }. We use the instances of the same concept in the Microsoft Concept Graph to construct the concept embedding. We assume that the embedding of a concept in implicit space is similar to its word embedding. Therefore, the concept embedding v_c is defined to be equal to the average of the weights of the instance embeddings plus the average of the relational representations as follows:

$$v_c = \frac{\sum_{i=1}^{k} w_i e_i}{\sum_{i=1}^{k} w_i} + \frac{\sum_{i=1}^{k} (e_c - e_i)}{n}. \quad (6)$$

Where the vector e_c of concepts and the vector e_i of instances are obtained by BERT embedding.

The rich concept information obtained from the Microsoft Concept Graph can make it easier for the machine to accomplish special tasks. Given the input sentence, we extract the entities from the sentence, and then take out the top k concepts corresponding to the entities based on the *Rep(e,c)* values, forming the concept set C, which is denoted as (v_{c1},v_{c2},...,v_{cm}), where $v_{c_i} \in \check{} \, ^{d_2}$ refers to the concept embedding calculated from Formula (6). We aim to generate the concept set representation p. Here, we introduce two attention mechanisms for generating weights for concepts to measure the importance of concepts.

The ambiguity of the entities and the noise in the knowledge base can cause the extraction of concepts that are not relevant to the text. For example, given the sentence, "Apple removes Wordle clones from the app store." The entity, "apple" in the Microsoft Concept Graph corresponds to two different meanings: *company* and *fruit*. There are also noise concepts, such as *juice*. Therefore, we introduce the text-to-concept attention mechanism (TCAM) to measure the similarity between concept vector v_{ci} and sentence representation s, which is used to select text-relevant concepts. Formally, TCAM is computed as follows:

$$\alpha_i = soft\max(v_{c_i}^T W_1 s + b_1) . \quad (7)$$

where α_i represents the attention weight of the *ith* concept in the concept set to the input sentence. A larger α_i indicates that the *ith* concept is more similar to the semantics of the sentence. We select concepts that are more similar to the sentence for ambiguous entities in this way, i.e., we assign larger weights to concepts that are more semantically similar to the sentence and smaller weights to concepts that do not match the semantics. Here, $W_1 \in \check{} \, ^{d_2 \times d_1}$ is a learnable parameter matrix, and b_1 is the offset. The softmax function is used to normalize attention weights.

Meanwhile, the importance of entities to the whole sentence is of great value for measuring the importance of concepts. Entities are the connection between text and concepts. For sentence classification, each entity has a different level of importance in the sentence, and the level of importance can also affect the importance of each concept in the concept set. For example, given the sentence, "Volkswagen falls further behind Tesla in the race to electric", we can identify the entities of "Volkswagen," "Tesla" and "electric". Obviously, "Volkswagen" and "Tesla" are more important than "electric" for classifying the sentence into the correct category "autos." Then, the concepts *automaker*, *brand*, *electric vehicle* corresponding to "Volkswagen" and "Tesla" should be correspondingly assigned greater weights in the whole concept set of {*automaker*, *brand*, *electric vehicle*, *utility*, *utility line*}. We use a self-attention mechanism to measure the importance of each entity to the sentence, and then normalize this importance score and assign it to the corresponding concept as follows:

$$\beta_i = \frac{\mathrm{softmax}(v^T \tanh(W_2 e_j + b_2))}{\sum_{e_k \in E} \mathrm{softmax}(v^T \tanh(W_2 e_k + b_2))} . \quad (8)$$

Where E is the entity set extracted from the sentence, $W_2 \in \check{} \, ^{d_b \times d}$ is a weight matrix, $v \in \check{} \, ^{d_b}$ is a weight vector, and b_2 is the offset. Meanwhile, we design a soft switch to dynamically combine α_i and β_i to obtain the final weight η_i of each concept as follows:

$$\eta_i = \gamma \alpha_i + (1-\gamma)\beta_i . \quad (9)$$

Where γ is an adjustable soft switch to adjust the importance of α_i and β_i.

Ultimately, the semantic representation c of the concept set is calculated by summing the weights of each concept embedding as follows:

$$c = \sum_{i=1}^{m} \eta_i v_{c_i} . \quad (10)$$

3.4. Output layer and loss function

We take concatenation with a residual connection [37] to integrate the representations of the sentence representation s and the concept set representation c. Therefore, we obtain a new mixed representation $s' \in \check{} \, ^{d_1}$ as follows:

$$s' = \tan h(W_3 concat(s,c) + b_3) + s . \quad (11)$$

Where *concat(s,c)* denotes the concatenation operation. $W_3 \in \check{} \, ^{d_2 \times (d_1 + d_2)}$ is a learnable weight matrix, and $b_3 \in \check{} \, ^{d_2}$ is an offset.

The representations are input to the softmax layer to calculate the conditional probability distributions over predefined categorical labels. We take the category cross-entropy loss as the training loss function. The formula for the overall loss value is as follows:

$$Loss = -\frac{1}{N} \sum_{i=1}^{N} \sum_{C=1}^{M} y_{ic} \log(p_{ic}) . \quad (12)$$

Where p_{ic} is the classification probability of the model. y_{ic} is the ground-truth value. C represents the label, M represents the total number of labels, i represents the sample, and N represents the total number of samples.

4. Experiments

4.1. Dataset

As shown in Table 1, we employ five public datasets to demonstrate the effectiveness of the proposed method. These datasets are public and available. We introduce them below:

AG's News[1]: The AG's News topic classification dataset is constructed by choosing the four topics from the original corpus. Each class contains 30,000 training samples and 1,900 testing samples. The total number of training samples is 120,000 and testing 7,600.

Yahoo! Answers[2]: The "Yahoo! Answers" contain 10 topics. It contains 140,000 training samples and 6,000 test samples. Each entry in the Yahoo! Answers dataset may contain two short questions and one longer answer. We concatenate the two question sentences together as input to our model.

MIND[3]: The Microsoft News Dataset (MIND) is a large-scale dataset for news recommendation research. It was collected from anonymized behavior logs of the Microsoft News website. MIND contains approximately 160,000 English news articles and more than 15 million impression logs generated by 1 million users. Every news article contains rich textual content, including a title, abstract, body, category, and entities.

DBpedia[4]: The DBpedia dataset is constructed from 14 topics selected from DBpedia 2014, including 40,000 training samples and 5,000 testing samples.

TREC-6[5]: TREC-6 is a sentence-level question classification dataset. It includes 6 topics, such as open domain and fact-based problems.

Table 1. Details of the experimental datasets

Datasets	#train	#test	#topic
AG's News	120K	7.6K	4
Yahoo! Answers	1.4M	60K	10
MIND	120K	30K	20
DBpedia	560K	70K	14
TREC-6	5.9K	500	6

4.2. Settings and Metrics

The proposed model uses AdamW optimizer for training. To stabilize training, we use the value 5e-5 to initialize the learning rate. We set the batch size as 64 and the training epochs as 20. We use pretrained 768-dimensional SBERT embeddings [24] to initialize word embeddings, and we fine-tune them in the training stage. All of the algorithms are implemented with PyTorch. For the LSTM, we found that the 256-dimensional hidden layer size obtains the best results. We use accuracy to evaluate the performance of the models. Accuracy is the probability of a correct prediction, i.e., the ratio of the number of samples correctly classified by the classifier to the total number of samples for a given test dataset.

4.3. Compared Method

To demonstrate the effectiveness of our proposed model, we chose some competitive models for comparison. The models are introduced below:

• BoW+SVM [38]: This model uses uni-gram as features of the text and then uses SVM as a classifier. This is the strong baseline of traditional text classification methods.

• VDCNN [39]: This model is much deeper than previously published convolutional neural networks and

[1] https://deepai.org/dataset/ag-news
[2] https://www.kaggle.com/datasets/jarupula/yahoo-answers-dataset
[3] https://msnews.github.io/
[4] https://www.dbpedia.org/
[5] https://cogcomp.seas.upenn.edu/Data/QA/QC/

operates directly at the character level through the constructed very small convolutions and pooling layers.

• Char-CNN [40]: This model is the first character-level convolutional network. In this model, only six hand-designed CNN layers were used. So it can achieve a very fast running time.

• Discriminative LSTM [41]: This model is based on the conventional LSTM with logistic regression and is a word-level model.

• KPCNN [11]: This model exploits a convolutional neural network for classification based on character and word level representations of concepts and texts. This model first conceptualizes texts as sets of relevant concepts through a large taxonomy knowledge base. Then, it coalesces the words and relevant concepts on top of pretrained word vectors to obtain the embedding of sentences. In addition, this model also incorporates the character-level feature to extract fine-grained information.

• DE-CNN [8]: This model uses a two-layer CNN to extract the context and conceptual information of the sentence text separately, and uses an attention mechanism to assign higher weights to the contextually relevant concepts.

• ULMFiT [42]: This model also uses multiple novel fine-tuning techniques that prevent catastrophic forgetting and enable robust learning across a diverse range of tasks.

• BERT-MLP [43]: This model uses two components that train each other jointly. One is a label denoiser, which estimates source reliability to reduce label noise on the matched samples. The other is a neural classifier, which predicts all of the samples and learns distributed representations. These two components are integrated into a co-training framework to benefit from each other

Table 2. Accuracy results of all methods. Our model is operated 10 times and reported by the mean and standard deviation. "—" indicates that it is not reported, and the best results are bolded.

Model	AG's News	Yahoo!Answers	MIND	DBpedia	TREC-6
BoW+SVM	0.727	0.692	—	0.967	—
VDCNN	0.913	0.734	0.902	0.987	—
Char-CNN	0.872	0.712	0.865	0.983	0.76
D-LSTM	0.921	0.737	0.914	—	—
KPCNN	0.883	0.725	0.902	0.987	0.934
DE-CNN	0.889	—	0.907	—	0.946
ULMFiT	0.923	0.739	0.922	0.986	0.964
BERT-MLP	0.925	0.742	0.923	0.988	0.969
CKAN (Our model)	**0.9432±0.0012**	**0.7684±0.0007**	**0.9361±0.0011**	**0.9931±0.0013**	**0.9741±0.0015**

4.4. Result

As shown in Table 2, the accuracy of our model is compared with that of the baseline model based on the five public datasets. The mean and the standard deviation of our model's accuracy are obtained by testing 10 times on each of the datasets. We find that the BoW+SVM model has the worst performance in classification. This is because this traditional approach uses the bag-of-words method to extract features, which is a statistical approach. However, the sentence text contains little content, and thus, it cannot provide sufficient statistical information. Additionally, there is a problem of data sparsity, so the classification effect is poor.

Distributed representation-based models, such as CNNs, LSTMs and pretrained language models, use low-dimensional, coherent and dense word vectors to represent text, which can effectively solve the problem of data sparsity compared with traditional methods. Therefore, the classification accuracy of these models is usually better than that of traditional methods. At the same time, the performance of the CNN has substantially improved compared with traditional methods because CNNs can capture different kinds of features using different convolutional kernels and pass the features to the pooling layer, which extracts salient features to effectively represent the text. Char-CNN represents the text by extracting character-level features; however, it does not perform well in these datasets. This is because the character-level features lose the semantic information of words in the text, and the small amount of content in a sentence makes it difficult for the model to capture the semantics of text through intercharacter relationships alone. VDCNN also extracts character-level features of sentence text, it is more effective than Char-CNN because VDCNN constructs a very deep CNN structure to extract more important feature information from the sentences. KPCNN enriches the semantics of sentence embedding with the introduction of conceptual knowledge into the sentence text representation.

In Table 2, we can see that it has better classification effectiveness than that of Char-CNN and VDCNN. This can indicate that the concept information can be used as a kind of prior knowledge to enhance the performance of a CNN in sentence-level classification. In addition, it was found that the accuracy of DE-CNN in several datasets has improved compared with VDCNN. This is because DE-CNN uses an attention mechanism to select text-relevant concepts and incorporate them into the sentence representation. Compared with KPCNN, it can reduce the impact of text-irrelevant concepts and noisy concepts on the performance of the classifier. A discriminative LSTM is better than CNNs because CNNs can only extract local features of text. In contrast, an LSTM is well-suited to handle text sequence information because of its ability to learn the current text information and the text information of the previous moment.

The classification effect of ULMFiT is better than that of CNNs and LSTMs because it adapts transfer learning. It conducts pretraining in a general corpus to learn general language knowledge, and then adapts fine-tuning in specific tasks. BERT also uses a pretraining and a fine-tuning paradigm, and has a stronger feature extractability than other methods because it uses a multilayer bidirectional transformer [44] to extract contextual features.

In Table 2, we can see that our model achieves the best classification performance on all five datasets. Compared with a CNN-based or RNN-based deep learning model, our model uses SBERT to extract features in the sentence, and thus, it has a stronger feature extractability. Meanwhile, compared with BERT-MLP, our model does not simply use the output of [CLS] as the embedding of the sentence. Instead, we use SBERT to obtain the word embedding of the sentence. At the same time, we introduce an LSTM to extract the contextual features, which can obtain a more effective sentence representation. Moreover, we use the conceptual knowledge from the additional knowledge base to extend the sentence and enrich the sentence semantics. In addition, we introduce two attention mechanisms, TCAM and ECAM, to assign weights to concepts. We also design a soft-switch method to adjust the ratio of these two weights to achieve the optimal classification performance.

4.5. Ablation Study

The main contribution of this paper is to introduce two attention mechanisms, TCAM and ECAM, to assign weights to concepts. We design a soft-switch mechanism to dynamically combine the two attention weights. To demonstrate the effectiveness of these contributions, we train and test the proposed model with its variants for comparison. Specific results are shown in Table 3 and Table 4.

Table 3. The influence of different modules on MIND dataset

	R_{base}	R_a	R_b	R_c	R_d
Concept knowledge		√	√	√	√
ECAM			√		
TCAM				√	
ECAM+TCAM					√
Accuracy	0.9241	0.9318	0.9331	0.9344	0.9357

Table 4. The influence of different modules on Yahoo! Answers dataset

	R_{base}	R_a	R_b	R_c	R_d
Concept knowledge		√	√	√	√
ECAM			√		
TCAM				√	
ECAM+TCAM					√
Accuracy	0.7426	0.7586	0.7619	0.7631	0.7658

We set R_{base} for the basic baseline. In this case, the baseline model only contains text encoding. As seen in Table 3 and Table 4, it was found that the performance of R_{base} only reaches 0.9241 accuracy on the MIND dataset and 0.7426 on the Yahoo! Answers dataset. Then, we incorporate the conceptual knowledge into the sentence representation. In this case, the concept set vector is simply concatenated with the sentence vector, and then fed into a fully connected layer for classification. The performance of R_a reaches 0.9318 accuracy on the MIND dataset and 0.7586 on the Yahoo! Answers dataset. This indicates that the sentence text lacks sufficient useful information for text classification due to the limitation of sentence length. With the aid of the Microsoft Concept Graph, we can enrich the representation of the sentence with concept information to improve the performance of the model. Although concept information can improve the performance of the model, the degree of importance of each concept in the concept set is consistent.

Therefore, we introduce the ECAM into the representation of the concept set. R_b means that we only use ECAM. We find that using ECAM to assign weights to concepts is more accurate than using R_a on both datasets. Because ECAM is able to assign greater weights to concepts corresponding to entities that are more important for classification. R_c means that we only use TCAM. TCAM can give more weight to the most context-relevant concepts. We find that the accuracy of R_c is improved compared with R_a for each of the two datasets. This indicates that contextual information has an important influence on the selection of concepts. R_d means that we use two attention mechanisms together, as well as a soft switch to adjust the ratio of the two attention weights at the same time.

According to Tables 2 and 3, it was found that R_d has higher accuracy than the previous methods on both datasets. This demonstrates that using soft switches to adjust the weights of the two attention mechanisms and reassigning weights to concepts can effectively improve the accuracy of the text classification model. This study demonstrates that our approach can make full use of conceptual knowledge for sentence classification, and that the contributions are effective.

4.6. Hyperparameter Adjustment

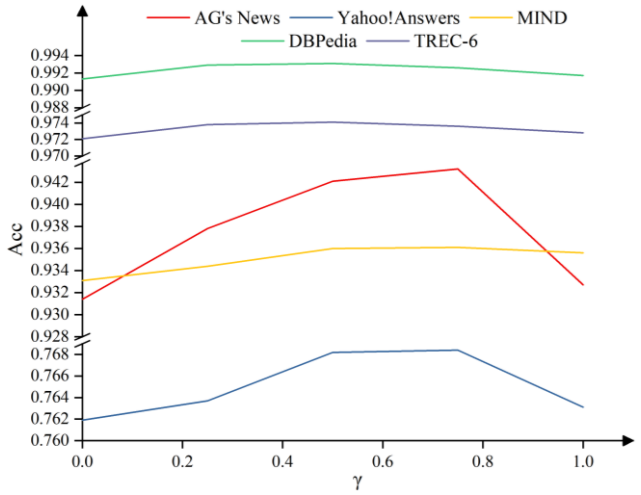


Figure 2. The effect of different hyperparameters γ on model performance

Fig. 2 shows the accuracy comparison of our model on five datasets using different values of γ. γ is in the interval of 0.25, ranging from 0 to 1. When γ = 0, only TCAM is used, and when γ = 1, only ECAM is used. We find that in general, the model accuracy is the highest when γ =0.75. However, the optimal values of γ on different datasets are different.

4.7. Power of Concepts

We incorporate conceptual information into sentence representation to improve the performance of sentence classification. To verify the power of concepts in our model, we selected several examples from the testing datasets to illustrate in Fig 3. These examples are assigned to the wrong labels in the traditional neural network, but our model can assign them into the correct labels.

Text : All eyes on Drew Brees as possibility of return nears.

Concepts : nfl player quarterback athlete

Text : Garth Brooks is playing a South Jersey spot as part of his Dive Bar tour. Next up? A football arena.

Concepts : artist country music singer musician

Figure 3. The two examples in MIND dataset. Underlined phrases are the entities, and the topic labels of the two sentences are sports and music respectively

When we classify the sentence text, there is a lack of context due to the short length. At the same time, the entities in the test examples may do not appear in the training dataset. It is difficult to classify them into the correct categories using traditional deep neural network models. However, when we introduce conceptual information, our model finds the corresponding concepts in the knowledge base to assist in classification. For example, in Fig.3, “Garth Brooks” is a rare word. It does not appear in the training dataset, so it is difficult to construct a representation for this entity using traditional models. The words "playing", "football" in the sentence also make it easy for traditional classifiers to misclassify the sentence into sports. However, our model can enrich sentence representation with concepts from the knowledge base to assist sentence classification.

4.8. Concept Embedding in CKAN

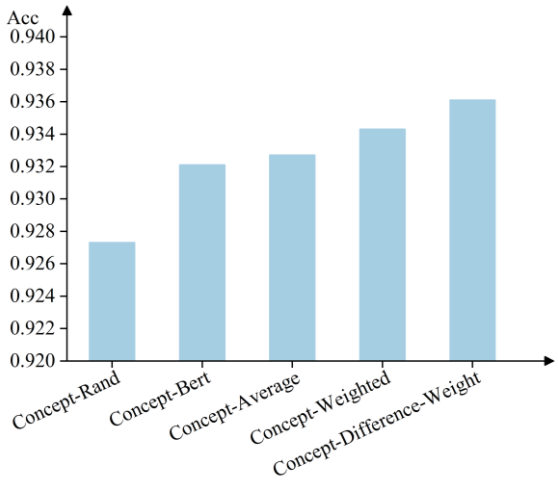
In our study, we propose a new multi-word concept embedding method. We compare the accuracy of our concept embedding method with the other four methods on two datasets. The descriptions of our concept embedding method and the methods compared are as follows:

- Concept-Rand: Concepts are randomly initialized and fine-tuned in the training stage.
- Concept-Bert: Concepts are first encoded in Word Piece, and then sent to the pretrained-BERT to obtain the concept representation.
- Concept-Instance-Average: As in Formula (13), concept embeddings are represented by the average of instance embedding, where instances are represented by BERT embedding.
- Concept-Instance-Weight-Average: As in Formula (14), concept embeddings are represented by the average of the weights of the instance embedding, where the weights of the instances are obtained from the Microsoft Concept Graph instantiation.

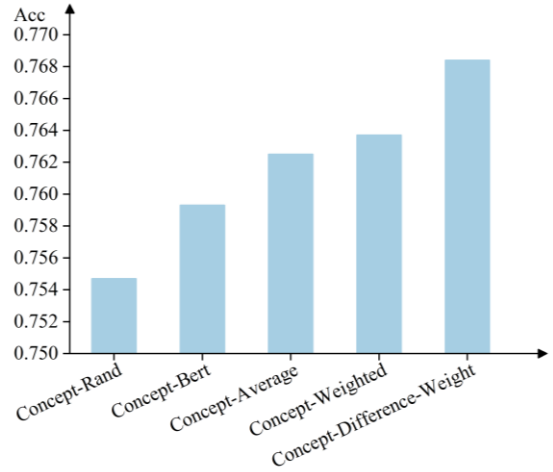
Concept-Instance-Difference-Weight-Average: Our proposed method, i.e., Formula (6)

$$v_c = \frac{\sum_{i\in(1,n)} e_{r_i}}{n} . \quad (13)$$

$$v_c = \frac{\sum_{i\in(1,n)} w_i e_{r_i}}{\sum_{i\in(1,n)} w_i} . \quad (14)$$



(a) Test on the MIND dataset



(b) Test on the Yahoo! Answers dataset

Figure 4. Comparison of the model's performance with different concept embedding methods

Fig. 4 shows the impact of different concept embedding methods on our model's performance on the two datasets. We find that Concept-Rand has the lowest accuracy. This is probably because the randomly initialized concept embedding requires large amounts of training resources to train, but our dataset is not large enough to train the concept embedding adequately. Concept-Bert has a significant improvement in accuracy over Concept-Rand. This is because BERT has gained general language knowledge, and it uses WordPiece to deal with OOV problems effectively. In the Microsoft Concept Graph a concept usually corresponds to multiple instances. Instances with the same meaning are often close in the implicit space. Therefore, it is better to use word vector averaging or weight averaging of instances to express the semantics of concepts than to only use deep learning models for estimation. However, the use of instance vector averaging does not accurately represent the true concept semantics because the difference between concept vectors and instance vectors is not considered. Our method can achieve the highest accuracy. This is because our method not only uses rich instances in the knowledge base to generate concept embeddings but also considers the differences between concept vectors and instance vectors to generate more accurate concept embeddings.

5. Conclusion and future work

In this paper, we propose a concept-kernel attention network. It contains two attention mechanisms for limiting the importance of contextually irrelevant concepts as well as noisy concepts, and then assigns greater weight to concepts that are important for classification. Meanwhile, we design a multi-word concept representation method that combines concept and entity relevance to obtain more accurate multi-word concept representation. Comparison experiments with strong baselines and ablation experiments demonstrate the effectiveness of CKAN.

In future work, we will try to incorporate conceptual information into the label embedding to enhance the semantic matching between text and labels. For example, we can construct a heterogeneous graph by counting the co-occurrence of concepts and labels in the training set. Then, we can use graph neural networks to obtain a label representation that incorporates the semantics of the relevant concepts.

Acknowledgments.

This work is supported by the National Natural Science Foundation of China (No. 62066009), the Key Research and Development Project of Guilin (No. 2020010308).

References

[1] Liu Y, Ji L, Huang R, Ming T, Gao C, Zhang J. An attention-gated convolutional neural network for sentence classification. Intelligent Data Analysis. 2019;23(5):1091-107.

[2] Türker R. Short text categorization using world knowledge. Karlsruhe: Karlsruher Institut für Technologie; 2021.

[3] Song G, Ye Y, Du X, Huang X, Bie S. Short text classification: A survey. Journal of multimedia. 2014;9(5):635-43.

[4] Yin J, Tang M, Cao J, Wang H, You M, Lin Y. Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning. World Wide Web. 2022;25(1):401-423.

[5] Zhang F, Wang Y, Liu S, Wang H. Decision-based evasion attacks on tree ensemble classifiers. World Wide Web. 2020; 23(5):2957-2977.

[6] Ge Y-F, Orlowska M, Cao J, Wang H, Zhang Y. MDDE: multitasking distributed differential evolution for privacy-preserving database fragmentation. VLDB J. 2022.

[7] Xie Q, Huang J, Peng M, Zhang Y, Peng K, Wang H. Discriminative Regularized Deep Generative Models for Semi-Supervised Learning. 2019 IEEE International Conference on Data Mining (ICDM); Nov. 8-11, 2019; Beijing, China: IEEE; 2019. pp. 658-667.
[8] Xu J, Cai Y, Wu X, Lei X, Huang Q, Leung H-f, Li Q. Incorporating context-relevant concepts into convolutional neural networks for short text classification. Neurocomputing. 2020;386:42-53.
[9] Liu Y, Li P, Hu X. Combining context-relevant features with multi-stage attention network for short text classification. Computer Speech & Language. 2022;71:101268.
[10] Wang F, Wang Z, Li Z, Wen J-R, editors. Concept-based Short Text Classification and Ranking. Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management; 2014 November 3-7; Shanghai, China: Association for Computing Machinery.
[11] Wang J, Wang Z, Zhang D, Yan J, editors. Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. IJCAI; 2017.
[12] Chen J, Hu Y, Liu J, Xiao Y, Jiang H, editors. Deep short text classification with knowledge powered attention. Proceedings of the AAAI Conference on Artificial Intelligence; 2019.
[13] Tao S, Sakai T, editors. Improving concept representations for short text classification. Proceedings of the 26th Annual Meeting of the Association for Natural Language Processing; 2020.
[14] Li P, Mao K, Xu Y, Li Q, Zhang J. Bag-of-Concepts representation for document classification based on automatic knowledge acquisition from probabilistic knowledge base. Knowledge-Based Systems. 2020;193:105436.
[15] Wu W, Li H, Wang H, Zhu KQ, editors. Probase: a probabilistic taxonomy for text understanding. Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data; 2012 May 20-24; Scottsdale, Arizona, USA: Association for Computing Machinery.
[16] Wu F, Qiao Y, Chen J-H, Wu C, Qi T, Lian J, Liu D, Xie X, Gao J, Wu W, editors. Mind: A large-scale dataset for news recommendation. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; 2020: Association for Computational Linguistics.
[17] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781. 2013.
[18] Pennington J, Socher R, Manning CD, editors. Glove: Global vectors for word representation. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP); 2014.
[19] Li W, Li L, editors. Combining Knowledge with Attention Neural Networks for Short Text Classification. International Conference on Knowledge Science, Engineering and Management; 2021; Berlin, Heidelberg: Springer, Cham.
[20] Jiang H, Yang D, Xiao Y, Wang W. Understanding a bag of words by conceptual labeling with prior weights. World Wide Web. 2020;23(4):2429-47.
[21] Devlin J, Chang M-W, Lee K, Toutanova K, editors. Bert: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019.
[22] Zhan J, Liao X, Bao Y, Gan L, Tan Z, Zhang M, He R, Lu J. An effective feature representation of web log data by leveraging byte pair encoding and TF-IDF. Proceedings of the ACM Turing Celebration Conference-China; May 17-19; Chengdu, China: Association for Computing Machinery; 2019. p. Article 62.
[23] Moens MF, Huang X-J, Specia L, Yih W-t, editors. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Proceedings of the Conference; 2021 November 7-11: The Association for Computational Linguistics.
[24] Reimers N, Gurevych I, editors. Sentence-bert: Sentence embeddings using siamese bert-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing; 2019.
[25] Li X, Li C, Chi J, Ouyang J. Short text topic modeling by exploring original documents. Knowl Inf Syst. 2018; 56(2):443-462.
[26] Rashid J, Shah SMA, Irtaza A. Fuzzy topic modeling approach for text mining over short text. Inform Process Manag. 2019; 56(6):102060.
[27] Gao W, Peng M, Wang H, Zhang Y, Xie Q, Tian G. Incorporating word embeddings into topic modeling of short text. Knowl Inf Syst. 2019; 61(2):1123-1145.
[28] Gao W, Peng M, Wang H, Zhang Y, Han W, Hu G, Xie Q. Generation of topic evolution graphs from short text streams. Neurocomputing. 2020; 383:282-294.
[29] Zhou P, Qi Z, Zheng S, Xu J, Bao H, Xu B. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. arXiv preprint arXiv:161106639. 2016.
[30] Peng M, Liao Q, Hu W, Tian G, Wang H, Zhang Y. Pattern Filtering Attention for Distant Supervised Relation Extraction via Online Clustering. In: Cheng R, Mamoulis N, Sun Y, Huang X, editors. Web Information Systems Engineering–WISE 2019. Cham: Springer International Publishing; 2019. pp. 310-325.
[31] Flisar J, Podgorelec V, editors. Document Enrichment using DBPedia Ontology for Short Text Classification. Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics; 2018 June 25-27; Novi Sad, Serbia: Association for Computing Machinery.
[32] Wang F, Wang Z, Li Z, Wen J-R. Concept-based Short Text Classification and Ranking. Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management; November 3-7, 2014; Shanghai, China: Association for Computing Machinery; 2014. pp. 1069-1078.
[33] Flisar J, Podgorelec V. Improving short text classification using information from DBpedia ontology. Fundamenta Informaticae. 2020;172(3):261-97.
[34] Ji L, Wang Y, Shi B, Zhang D, Wang Z, Yan J. Microsoft concept graph: Mining semantic concepts for short text understanding. Data Intelligence. 2019;1(3):238-70.
[35] Wang Z, Wang H, Wen J-R, Xiao Y, editors. An Inference Approach to Basic Level of Categorization. Proceedings of the 24th ACM International on Conference on Information and Knowledge Management; 2015 October 18-23; Melbourne, Australia: Association for Computing Machinery.
[36] Qi P, Zhang Y, Zhang Y, Bolton J, Manning CD, editors. Stanza: A python natural language processing toolkit for many human languages. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations; 2020.
[37] Hao Y, Zhang Y, Liu K, He S, Liu Z, Wu H, Zhao J, editors. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics; 2017.

[38] Wang SI, Manning CD, editors. Baselines and bigrams: Simple, good sentiment and topic classification. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics; 2012.
[39] Conneau A, Schwenk H, Barrault L, Lecun Y, editors. Very deep convolutional networks for text classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics; 2016: Association for Computational Linguistics.
[40] Zhang X, Zhao J, LeCun Y, editors. Character-level convolutional networks for text classification. Advances in Neural Information Processing Systems; 2015.
[41] Yogatama D, Dyer C, Ling W, Blunsom P. Generative and discriminative text classification with recurrent neural networks. arXiv preprint arXiv:170301898. 2017.
[42] Howard J, Ruder S. Universal language model fine-tuning for text classification. arXiv preprint arXiv:180106146. 2018.
[43] Ren W, Li Y, Su H, Kartchner D, Mitchell C, Zhang C, editors. Denoising multi-source weak supervision for neural text classification. Findings of the Association for Computational Linguistics: EMNLP 2020; 2020.
[44] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I, editors. Attention is all you need. Advances in Neural Information Processing Systems; 2017.