Knowledge Powered Cooperative Semantic Fusion for Patent Classification

Zhe Zhang[1], Tong Xu[1(✉)], Le Zhang[1], Yichao Du[1], Hui Xiong[2(✉)], and Enhong Chen[1]

[1] School of Computer Science and Technology, University of Science and Technology of China, Hefei, China
{tongxu,cheneh}@ustc.edu.cn, duyichao@mail.ustc.edu.cn
[2] The State University of New Jersey, New Brunswick, USA
hxiong@rutgers.edu

Abstract. Patent classification is beneficial for many patent applications, such as patent quality valuation, retrieval, and litigation analysis. Recently, many automatic patent classification methods have been proposed to save labor costs, which usually formulate this task as a multi-label text classification problem. In reality, patent language is highly terminological, full of scientific entities and domain knowledge. However, existing works seldom consider such unique property of patents, which reduces the classification performance. To this end, we propose a novel framework named Knowledge Powered Cooperative Semantic Fusion to capture deeper knowledge semantics for patent classification. Specifically, we first exploit knowledge graphs to enrich the patent with related entities. Then we design a mutual attention mechanism between entities and original texts to emphasize the crucial semantics of entities with the guide of texts, and vice versa. Finally, we introduce the graph convolutional network further to enhance the fusion representation of entities and texts. Extensive experiments on large-scale patent data demonstrate the superior performance of our model on the patent classification task.

Keywords: Patent classification · Knowledge graph · Attention mechanism · Graph convolutional network

1 Introduction

Patent classification is regarded as a basic task in the field of patent management, which can provide support for many downstream intelligent tasks, such as patent quality valuation [1], patent retrieval [2], and patent litigation analysis [3]. To avoid the ambiguity, patent classification schemes such as International Patent Classification (IPC[1]) and Cooperative Patent Classification (CPC[2]) are proposed to standardize patent categories. For instance, in CPC scheme, code

[1] https://www.wipo.int/classifications/ipc/en/.
[2] https://www.cooperativepatentclassification.org/index.

L. Fang et al. (Eds.): CICAI 2021, LNAI 13069, pp. 111–122, 2021.
https://doi.org/10.1007/978-3-030-93046-2_10

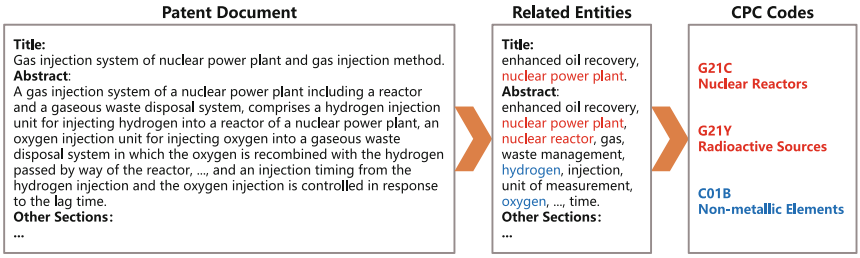

Fig. 1. A toy example of a patent document and its related entities.

"G06F 40/20" refers to category "Natural Language Analysis" and "G06F 40/56" refers to "Natural Language Generation". Traditionally, patent classification is completed by well-trained specialists, which is labor-intensive and sometimes error-prone because the code system is vast and growing.

Consequently, automatic patent classification has aroused widespread attention in the industry and academia [4,5]. Since the patent that contains title, abstract and other sections is usually long text and can be classified into multiple categories, most researchers have treated this task as a multi-label text classification problem. Concretely, shallow machine learning-based methods [6,7] usually focus on learning handcrafted feature combinations and deep learning-based methods [8,9] are dedicated to capturing the contextual semantics of patent texts. Although these methods have achieved great success by mining pure text semantics, they usually ignore the terminology of patents. Specifically, the patents are related to a large number of knowledge entities that can play an important role in patent classification. As shown in Fig. 1, it presents a patent, related entities discovered with entity linking technology [10,11], and CPC category codes of the patent. We can observe that the red entities are closely associated with code "G21C (Nuclear Reactors)" and "G21Y (Radioactive Sources)", while the blue ones are closely associated with "C01B (Non-metallic Elements)". In other words, these related entities can provide additional distinguishable semantics for patent classification besides original texts.

However, there are still many unique challenges in incorporating these entity semantics with pure text semantics into patent classification. First, it is difficult to mine such entity semantics with previous methods because the related entities may be very sparse in the patent corpus, which becomes the bottleneck of improving patent classification. Second, the importance of different entities to patent classification varies greatly, and domain-specific entities are usually more helpful. Take Fig. 1 as an example, "nuclear power plant" and "hydrogen" are strongly associated with target categories, while "gas" seems to be useless for classification. More seriously, due to the limitations of entity linking technology, some wrong entities may be introduced such as "enhanced oil recovery". Third, patent texts usually contain hundreds of words, but only a few key fragments can provide valuable information for classification. Extracting crucial fragments for target categories is as tricky as finding a needle in a haystack.

To address these challenges, we propose **K**nowledge Powered **C**ooperative **S**emantic **F**usion (KCSF) that jointly models the text semantics and entity semantics for more distinguishable representation of the patent. It achieves better performance on patent classification task by incorporating Knowledge Graphs (KG), mutual attention mechanism, and Graph Convolutional Network (GCN) [12]. The technical contributions of this paper are summarized as follows:

- We propose to employ entity linking and knowledge graph embedding techniques to introduce additional knowledge into semantic modeling so that the deeper entity semantics can be captured for patent classification.
- We design a novel mutual attention mechanism to extract the crucial semantics in texts with the help of entities and then reduce the bad influence of improper entities with the generated features of texts. Furthermore, we introduce the graph convolutional network to facilitate the fusion representation learning of texts and entities towards better classification performance.
- Extensive experiments on large-scale patent data clearly validate the effectiveness of our model, which also demonstrate the potentiality of knowledge-enhanced methods on patent classification task.

2 Related Work

2.1 Patent Classification

With the advances of natural language processing technology, many methods have been proposed to perform automatic multi-label patent classification such as KNN [13] and SVM [7]. These methods represent patent texts by contained words but ignore the contextual information and deep semantic information. To address this problem, deep learning techniques have been gradually applied on patent classification. For instance, based on TextCNN [14], DeepPatent [8] builds a deep convolutional neural network combined with the word embedding. BiGRU [15] is also used to encode patents based on domain-specific word embedding [4]. PatentBERT [9] utilizes pre-trained language model BERT [16] to represent the patent and then fine-tune it. In addition, A-GCN+A-NLSOA [5] attempts to study the patent classification with graph representation learning, which focuses on the links among patents and words. These methods usually take the original patent texts as input but ignore the scientific entities and common sense existing in patents, leading to limitations in their performance.

2.2 Knowledge-Enhanced Short Text Classification

Due to the lack of contextual semantics in short texts, researchers have gradually realized the importance of introducing knowledge as additional semantics [17,18]. Specifically, KPCNN [19] proposes to conceptualize the short texts as relevant concepts predefined in knowledge graphs and then stacks the words and concepts to obtain the embedding of the short texts. Based on that, STCKA [20] further introduces the attention mechanism to measure the importance of each

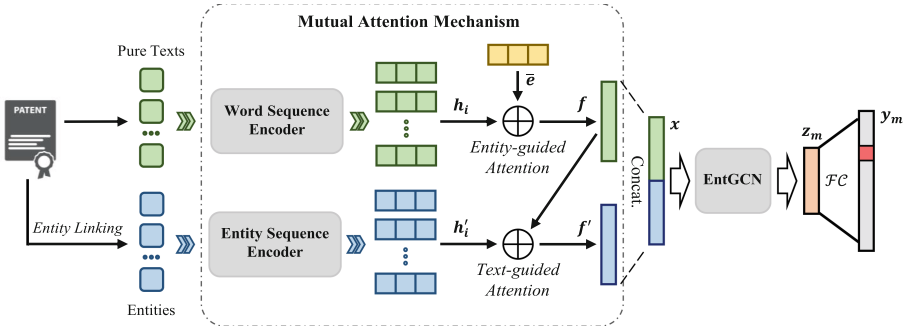

Fig. 2. The framework of KCSF for patent classification.

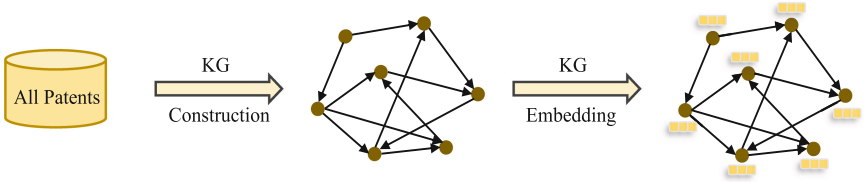

Fig. 3. Illustration of knowledge powered semantic augmentation process.

concept. Moreover, HGAT [21] attempts to model short texts, related entities, and contained topics simultaneously with heterogeneous information networks and adapts graph neural networks for semi-supervised classification. Inspired by these methods, we utilize knowledge graphs to enrich the semantic features of patents and design a dedicated cooperative semantic fusion framework.

3 The Proposed Model KCSF

Our model **KCSF** is a knowledge powered deep neural network as shown in Fig. 2. It consists of three key components: Knowledge Powered Semantic Augmentation, Mutual Attention Mechanism, and Entity-based Graph Convolutional Network (EntGCN).

3.1 Knowledge Powered Semantic Augmentation

The knowledge powered semantic augmentation aims at discovering and representing entities related to patent texts, which is shown in Fig. 3. First, entity linking tool TagMe[3] is used to recognize the entities in the patent texts. For example, in the patent title "Method for the contactless charging of the battery of an electric automobile", "charging" is linked with the entity "battery charger",

[3] https://sobigdata.d4science.org/web/tagme/.

while "electric" and "automobile" are linked with the entity "electric car". Second, based on all identified entities of all patents, we construct a sub-graph G by extracting all relations among them from KG DBpedia [22]. To enrich the relational information, we further expand G to all entities in the one-hop neighborhood of identified ones. Third, the KG embedding method TransE [23] is utilized to learn a low-dimensional embedding vector for each entity in G.

Specifically, the KG G consists of a large number of entity-relation-entity triples (h, r, t), where h, r, and t are the head entity, the relation, and the tail entity, respectively. TransE defines the score function as:

$$f_r(h, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2, \tag{1}$$

where $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$ are the corresponding embedding vector of h, r, and t. The goal of TransE is to force $f_r(h, t)$ to be low if (h, r, t) is true, and high otherwise. In this manner, the embedding of each entity can be trained to preserve both relational information and structural information, which can provide additional distinguishable features for patent classification.

For a patent composed of a sequence of words, i.e., $s = [w_1, w_2, ..., w_n]$, each word may be associated with an entity in the KG. So the patent can be also processed as a sequence of entities, i.e., $s' = [e_1, e_2, ...e_{n'}]$ and each entity e_i can be represented as a vector via KG embedding.

3.2 Mutual Attention Mechanism

We design the mutual attention mechanism to model the original texts and related entities jointly. Specifically, we first employ two sequence encoders on word sequence and entity sequence respectively to get the corresponding hidden features. Then we employ two types of attention mechanisms consecutively to enhance the representation of entities with the guide of texts and vice versa.

Word/Entity Sequence Encoder. We construct the word sequence encoder based on word2vec [24] and Bidirectional Gated Recurrent Unit (BiGRU) [15]. First, given the word sequence $s = [w_1, w_2, ..., w_n]$, each word is mapped to an embedding vector $\mathbf{w}_i \in \mathbb{R}^{d_w}$ via word2vec, where d_w denotes the size of word embedding. Then we utilize BiGRU to encode patterns in word sequence to get the hidden features. Specifically, the input of BiGRU is a word embedding sequence $\mathbf{s} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n]$, and the hidden feature $\mathbf{h}_i$ is calculated as follows:

$$\begin{aligned} \overrightarrow{\mathbf{h}_i} &= \overrightarrow{\mathrm{GRU}}(\overrightarrow{\mathbf{h}_{i-1}}, \mathbf{w}_i) & \overrightarrow{\mathbf{h}_i} &\in \mathbb{R}^{d_h}, \\ \overleftarrow{\mathbf{h}_i} &= \overleftarrow{\mathrm{GRU}}(\overleftarrow{\mathbf{h}_{i+1}}, \mathbf{w}_i) & \overleftarrow{\mathbf{h}_i} &\in \mathbb{R}^{d_h}, \\ \mathbf{h}_i &= [\overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}] & \mathbf{h}_i &\in \mathbb{R}^{2d_h}, \end{aligned} \tag{2}$$

where d_h is the hidden size of GRU and the semicolon refers to concatenation.

Next, we adopt a similar architecture to construct the entity sequence encoder. Given the entity sequence $s' = [e_1, e_2, ..., e_{n'}]$, each entity is mapped to

an embedding vector $\mathbf{e}_i \in \mathbb{R}^{d_e}$ via TransE, where d_e is the size of entity embedding. Then we employ another BiGRU (i.e., $\overrightarrow{\text{GRU}'}$ and $\overleftarrow{\text{GRU}'}$) to get the hidden feature $\mathbf{h}'_i \in \mathbb{R}^{2d_h}$ for each entity. The detailed formula is similar to Eq. 2.

Entity-Guided Attention. Afterwards, we try to use the entities to enhance the representation of pure texts because the text semantics that are also reflected by entities are usually more crucial. To this end, we propose the entity-guided attention to evaluate the importance of different words with the help of entities. We first exploit the average pooling operation to merge n' entity embeddings into an average embedding $\bar{\mathbf{e}}$ and then feed it into vanilla attention [25] to calculate the attention weights α w.r.t each hidden feature $\mathbf{h}_i$ as follows:

$$\begin{aligned} \bar{\mathbf{e}} &= \frac{1}{n'} \sum_{i=1}^{n'} \mathbf{e}_i, \\ \alpha_i &= \text{softmax}(\mathbf{v}^{(1)} \cdot \tanh(\mathbf{W}^{(1)}[\mathbf{h}_i; \bar{\mathbf{e}}] + \mathbf{b}^{(1)})), \end{aligned} \tag{3}$$

where $\mathbf{W}^{(1)}$, $\mathbf{b}^{(1)}$, and $\mathbf{v}^{(1)}$ are trainable parameters. We combine all the hidden features of words to get the fusion feature $\mathbf{f}$ for word sequence as:

$$\mathbf{f} = \frac{1}{n} \sum_{i=1}^{n} \alpha_i \mathbf{h}_i. \tag{4}$$

Text-Guided Attention. To reduce the bad influence of improper entities introduced due to the complexity of patent language or the imprecision of entity linking, we further propose the text-guided attention as follows:

$$\alpha'_i = \text{softmax}(\mathbf{v}^{(2)} \cdot \tanh(\mathbf{W}^{(2)}[\mathbf{h}'_i; \mathbf{f}] + \mathbf{b}^{(2)})), \tag{5}$$

$$\mathbf{f}' = \frac{1}{n'} \sum_{i=1}^{n'} \alpha_i \mathbf{h}'_i, \tag{6}$$

where $\mathbf{f}$ is the fusion feature of word sequence in Eq. 4 and $\mathbf{h}'_i (i \in \{1, ..., n'\})$ is the hidden feature for each entity obtained by BiGRU. The motivation is that the entities with semantics that are not similar to text semantics are usually insignificant or even noise, which is also observed in work [20]. Finally, we concatenate the two types of fusion features of patents to get the joint fusion feature $\mathbf{x} \in \mathbb{R}^{4d_h}$, i.e., $\mathbf{x} = [\mathbf{f}; \mathbf{f}']$.

3.3 Entity-Based Graph Convolutional Network

Intuitively, for patents with similar entities, their scientific fields are usually very similar, so they may have similar categories. Along this line, to further emphasize the crucial information in both the texts and entities, we consider the relations among different patents. Specifically, for the target patent p_m, we first compute the entity-based similarity between it and other patents, defined as the cosine

similarity between their average entity embeddings, i.e., $\bar{\mathbf{e}}$ in Eq. 3. Next, we select the top K most similar patents of p_m and itself as the nodes and then compute the similarity scores between these $K+1$ patents as the weighted edges to constructed a neighborhood graph for p_m.

We use $\mathbf{A}$ to denote the adjacency matrix of the graph in which $\mathbf{A}(i,j)$ is the entity-based similarity between patent p_i and patent p_j. Let $\mathbf{D}$ denote the degree matrix. Moreover, we first obtain the joint fusion feature $\mathbf{x} \in \mathbb{R}^{4d_h}$ for each patent node and then stack them to get the feature matrix $\mathbf{X} \in \mathbb{R}^{(K+1)\times 4d_h}$. Because the graph only involves the one-hop neighbor nodes of p_m, we employ a single layer GCN [12] to enhance the presentation of p_m with its neighborhood:

$$\mathbf{Z} = \mathrm{ReLU}(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}^{(3)}). \tag{7}$$

Here, $\mathrm{ReLU}(\cdot)$ is rectified linear unit [26] and the m-th row of the output matrix $\mathbf{Z} \in \mathbb{R}^{(K+1)\times d_z}$ is the enhanced fusion feature of p_m. Let $\mathbf{z}_m \in \mathbb{R}^{d_z}$ denote the m-th row of $\mathbf{Z}$. We feed it into a fully connected layer for classification:

$$\hat{\mathbf{y}}_m = \mathrm{softmax}(\mathbf{W}^{(4)}\mathbf{z}_m + \mathbf{b}^{(3)}), \tag{8}$$

where $\hat{\mathbf{y}}_m$ is the predicted categories. Then we apply binary cross-entropy loss as the objective function, which is often used in multi-label text classification [27].

4 Experiments

4.1 Experimental Setup

Dataset and Evaluation Metrics. We built a dataset named USPTO-1M from the website of the United States Patent and Trademark Office (USPTO[4]), which has granted millions of USA patents since 1976. We first collected 1,441,172 patents from the USPTO website and only retained the patents containing both the title and abstract. Next, we adopted the exact same data cleaning process as DeepPatent [8], which included filtering low-frequency words, removing too short patents, etc. As a result, the USPTO-1M dataset contained 1,086,422 patents in 661 CPC subclass-level categories, and each patent had 1.88 categories averagely. We split the dataset into training and testing in an 80/20 ratio and further held 10% training data as the validation set to choose the optimal parameters.

We adopted the rank-based metrics including Precision@k, Recall@k, and NDCG@k (Normalized Discounted Cumulative Gain), which were widely used in multi-label text classification [5,28]. Particularly, we set k as 1, 3, and 5.

Implementation Details. For training KCSF, we used the Adam [29] optimizer and set the learning rate and weight decay to 1×10^{-3} and 5×10^{-5}, respectively. We set the dropout [30] probability to 0.4 and the batch size to 32. We also applied an early stop mechanism, in which the training would stop if

[4] www.uspto.gov.

the Precision@1 on the validation set did not improve in 10 continuous epochs. We trained word2vec [24] for word embeddings with $d_w = 100$ and trained TransE [23] for entity embeddings with $d_e = 100$. For the remaining parameters, we used the grid search for the optimal values. Specifically, we set $d_h = 256$ for the hidden states in BiGRU, $K = 8$ to construct the neighborhood graph, and $d_z = 384$ for the output of EntGCN.

Baselines. We compared our model KCSF with the following baselines, including general text classification models, patent classification models, knowledge-enhanced short text classification models and two variants of KCSF:

- FastText [31]: It is a widely used text classification model that makes full use of n-gram features for text representation.
- BiLSTM-SA [32]: It takes the benefit of BiLSTM and self-attention mechanism to mine deeper contextual semantics for classification.
- DeepPatent [8]: It is a deep learning-based patent classification model with core component based on the architecture of TextCNN [14].
- PatentBERT [9]: It applies BERT [16] to encode patent texts and classifies patents to multiple categories accurately by fine-tuning BERT.
- KPCNN [19]: It uses relevant concepts to enrich the semantics of short texts and adopts TextCNN to learn the coalesced embedding of concepts and texts.
- STCKA [20]: It is the state-of-the-art model for short texts classification, utilizing attention mechanism to evaluate the importance of each concept.
- KCSF-MAM: It is a variant of KCSF, without considering the mutual guidance between entities and texts. In other words, it uses the following formula to replace the mutual attention mechanism:

$$\mathbf{x} = [(\frac{1}{n}\sum_{i=1}^{n}\mathbf{h}_i); (\frac{1}{n'}\sum_{i=1}^{n'}\mathbf{h}'_i)]. \tag{9}$$

- KCSF-GCN: It is a variant of KCSF, which discards the relations among different patents by removing the EntGCN module from our model.

4.2 Experimental Results and Ablation Studies

Comparison Between Different Models. We concatenated the title and abstract of the patent together as the original texts input into different models and focused on the subclass-level categories defined by the CPC schema. According to the results shown in Table 1, we have the following observations:

- Although both STCKA and BiLSTM-SA are BiLSTM-based models with attention mechanism, STCKA achieves an improvement of 3.6% on Pricision@1 against BiLSTM-SA, which means that external knowledge can significantly improve the results of patent classification. The performance of KPCNN over DeepPatent, BiLSTM-SA, and FastText also proves this point.

Table 1. Results of multi-label patent classification on USPTO-1M.

Models	Precision@k (%)			Recall@k (%)			NDCG@k (%)		
	1	3	5	1	3	5	1	3	5
FastText	78.96	44.43	31.31	53.61	78.62	84.47	78.96	78.12	79.24
BiLSTM-SA	81.23	45.77	32.24	54.83	79.64	86.63	81.23	80.29	81.06
DeepPatent	81.38	45.93	32.48	54.80	79.86	86.53	81.38	80.66	81.41
PatentBERT	85.23	49.88	34.82	58.47	83.44	90.26	85.23	84.08	85.26
KPCNN	82.57	46.64	33.29	56.20	80.37	86.99	82.57	81.49	82.71
STCKA	84.78	49.21	34.73	57.49	83.22	89.16	84.78	83.66	85.29
KCSF	**87.82**	**51.27**	**36.76**	**59.91**	**84.23**	**91.74**	**87.82**	**86.04**	**87.73**

- The importance of different entities to patent classification varies greatly, and the attention mechanism can capture this difference well. Our model employs the mutual attention mechanism to evaluate the importance of different entities, and hence it performs much better than all baselines. On the contrary, KPCNN does not consider this issue, and thus it only performs a little bit better than another CNN-based model, i.e., DeepPatent.
- Our model obtains 3.0%, 2.1% and 2.0% improvements in precision and 3.0%, 2.4% and 2.4% in NDCG over STCKA. The reason is that our model is aware of the different roles of each fragment in patent texts and uses the entity-guided attention to emphasize the key fragments.
- PatentBERT outperforms all the other baselines because BERT can encode much more semantic information in word embeddings than common word2vec. KCSF still achieves better performance than PatentBERT, once again validating the effectiveness of cooperative semantic fusion.

Ablation Studies. We conducted ablation studies to evaluate the effectiveness of each module in our model KCSF, and the results are shown in Table 2. Particularly, KCSF-MAM ignores the harm of inappropriate entities and non-key words to the representation of patents, resulting in too much worthless semantic information being input into EntGCN. So its performance is much worse than KCSF. Moreover, KCSF-GCN discards the relations among patents, so that the fusion representation of texts and entities can not be further refined by aggregating additional semantic information from similar patents. That is the reason why KCSF-GCN performs worse than KCSF. In summary, the cooperation of our model is not only in the use of mutual attention mechanism to jointly model the entity semantics and text semantics, but also in the use of EntGCN to enhance each other between different patents.

4.3 Sensitivity Analysis on Neighborhood Graph Size

As mentioned in Sect. 3.3, the top K most similar patents of target patent p_m are selected to construct the neighborhood graph. In other words, K not only

Table 2. Ablation studies.

Models	Precision@k (%)			Recall@k (%)			NDCG@k (%)		
	1	3	5	1	3	5	1	3	5
KCSF-MAM	82.79	46.66	33.18	56.53	80.81	87.12	82.79	81.24	82.60
KCSF-GCN	85.49	49.76	34.95	58.50	83.69	90.14	85.49	84.55	85.54
KCSF	**87.82**	**51.27**	**36.76**	**59.91**	**84.23**	**91.74**	**87.82**	**86.04**	**87.73**

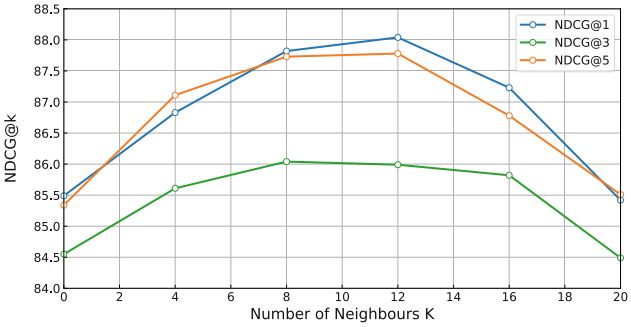

Fig. 4. Parameter sensitivity of KCSF.

determines the size of the graph but also reflects the quality of these neighbors. We tested all K in the set $\{0, 4, 8, 12, 16, 20\}$ by examining how they affect the performance of our model. According to Fig. 4, we realize that when $K < 12$, the performance keeps improving, but the improvement becomes more limited with larger K. Obviously, these similar patents can provide rich semantic information to enhance the representation of p_m. However, as more and more neighbors are considered, when a new neighbor is integrated, its contribution will be limited compared with the known information. More seriously, too large K may cause many patents that are not similar to p_m to be considered, resulting in a large amount of semantic noise being gathered by EntGCN into the representation of p_m. This is why the performance begins to deteriorate when $K > 12$.

5 Conclusion

In this paper, we proposed the KCSF framework to perform knowledge-enhanced patent classification. Specifically, we designed the mutual attention mechanism to capture the crucial semantics of entities with the guide of texts and vice versa. Moreover, we introduced the graph convolutional network to further enhance the fusion representation of entities and texts. Experimental results showed that our model had obtained substantial improvements on patent classification task.

Acknowledgement. This research was supported by the National Key Research and Development Program of China (Grant No. 2018YFB1402600), and the National Natural Science Foundation of China (Grant No. 91746301, 62072423).

References

1. Lin, H., Wang, H., Du, D., Wu, H., Chang, B., Chen, E.: Patent quality valuation with deep learning models. In: Pei, J., Manolopoulos, Y., Sadiq, S., Li, J. (eds.) DASFAA 2018. LNCS, vol. 10828, pp. 474–490. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91458-9_29
2. Fujii, A.: Enhancing patent retrieval by citation analysis. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 793–794 (2007)
3. Liu, Q., Wu, H., Ye, Y., Zhao, H., Liu, C., Du, D.: Patent litigation prediction: a convolutional tensor factorization approach. In: IJCAI, pp. 5052–5059 (2018)
4. Risch, J., Krestel, R.: Domain-specific word embeddings for patent classification. Data Technol. Appl. (2019)
5. Tang, P., Jiang, M., (Ning) Xia, B., Pitera, J.W., Welser, J., Chawla, N.V.: Multi-label patent categorization with non-local attention-based graph convolutional network. In: AAAI, pp. 9024–9031 (2020)
6. D'hondt, E., Verberne, S., Koster, C., Boves, L.: Text representations for patent classification. Comput. Linguist. **39**(3), 755–775 (2013)
7. Chih-Hung, W., Ken, Y., Huang, T.: Patent classification system using a new hybrid genetic algorithm support vector machine. Appl. Soft Comput. **10**(4), 1164–1177 (2010)
8. Li, S., Jie, H., Cui, Y., Jianjun, H.: DeepPatent: patent classification with convolutional neural networks and word embedding. Scientometrics **117**(2), 721–744 (2018)
9. Lee, J.-S., Hsiang, J.: Patent classification by fine-tuning BERT language model. World Patent Inf. **61**, 101965 (2020)
10. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518 (2008)
11. Sil, A., Yates, A.: Re-ranking for joint named-entity recognition and linking. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 2369–2374 (2013)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
13. Fall, C.J., Törcsvári, A., Benzineb, K., Karetka, G.: Automated categorization in the international patent classification. In: ACM SIGIR Forum, vol. 37, pp. 10–25. ACM, New York (2003)
14. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
15. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
16. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
17. Jingyun, X., et al.: Incorporating context-relevant concepts into convolutional neural networks for short text classification. Neurocomputing **386**, 42–53 (2020)
18. Alam, M., Bie, Q., Türker, R., Sack, H.: Entity-based short text classification using convolutional neural networks. In: Keet, C.M., Dumontier, M. (eds.) EKAW 2020. LNCS (LNAI), vol. 12387, pp. 136–146. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-61244-3_9

19. Wang, J., Wang, Z., Zhang, D., Yan, J.: Combining knowledge with deep convolutional neural networks for short text classification. In: IJCAI, vol. 350 (2017)
20. Chen, J., Yizhou, H., Liu, J., Xiao, Y., Jiang, H.: Deep short text classification with knowledge powered attention. In: Proceedings of the AAAI Conference on Artificial Intelligence vol. 33, pp. 6252–6259 (2019)
21. Linmei, H., Yang, T., Shi, C., Ji, H., Li, X.: Heterogeneous graph attention networks for semi-supervised short text classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 4823–4832 (2019)
22. Lehmann, J., et al.: DBpedia-a large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web **6**(2), 167–195 (2015)
23. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Neural Information Processing Systems (NIPS), pp. 1–9 (2013)
24. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, vol. 26, pp. 3111–3119 (2013)
25. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
26. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105 (2012)
27. Huang, W., et al.: Hierarchical multi-label text classification: an attention-based recurrent network approach. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1051–1060 (2019)
28. Prabhu, Y., Varma, M.: FastXML: a fast, accurate and stable tree-classifier for extreme multi-label learning. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 263–272 (2014)
29. Kingma, D.P., Adam, J.B.: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
30. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
31. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
32. Lin, Z., et al.: A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017)