Matching Document Pairs using Multi-Feature Semantic Fusion Based on Knowledge Graph

Yibo Chen
Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Zuping Zhang (✉ zpzhang@csu.edu.cn)
School of Computer Science and Engineering, Central South University, ChangSha, Hunan

Xin Huang
Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Xing Xiang
Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Zhiqiang He
State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Yunsheng Chen
Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Qihui Hu
Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan

Research Article

Keywords: Matching document pairs, Multi-feature semantic, Knowledge graph, Concept graph

Posted Date: October 12th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-907473/v1

Matching Document Pairs using Multi-Feature Semantic Fusion Based on Knowledge Graph

Yibo Chen [a] , Zuping Zhang[b,*], Xin Huang[a], Xing Xiang[a],Zhiqiang He[c], Yunsheng Chen[a], Qihui Hu[a]

[a]*Information and Communication Branch of State Grid Hunan Electric Power Company Limited, Changsha, Hunan, China*

[b] *School of Computer Science and Engineering, Central South University, Changsha, Hunan, China*

[c] *State Grid Hunan Electric Power Company Limited, Changsha, Hunan, China*

ARTICLEINFO

Keywords:
Matching document pairs
Multi-feature semantic
Knowledge graph
Concept graph

ABSTRACT

Discriminating the homology and heterogeneity of two documents in information retrieval is very important and difficult step. Existing methods mainly focus on word-based document duplicate checking or sentence pairs matching except manual verification which need a lot of human resource cost. The word-based document duplicate checking can not judge the similarity of two documents from the semantic level and the matching sentence pair methods can not effectively mine the semantic information from a long text which is frequent retrieval results.

A concept-based Multi-Feature Semantic Fusion Model (MFSFM) is proposed. It employs multi-feature enhanced semantics to construct a concept map for represent the document, and employs a multi-convolution mixed residual CNN module to introduce local attention mechanism for improve the sensitivity of conceptual boundary information. To improve the feasibility of the proposed MFSFM based on concept maps, two multi-feature document data sets are set up. Each of them consists of about 500 actual scientific and technological project feasibility reports. Experimental results based on the actual datasets show that the proposed MFSFM converges quickly while expanding the latest methods of natural language matching at the accuracy rate.

1. Introduction

Recognizing the relationship of document pairs is an indispensable Natural Language Understanding (NLU) task, which is essential for document duplication and document search. For example, a project system needs to review newly declared projects to check whether there are duplicate declarations. Early document recognition methods were based on term similarity and rules. Traditional matching methods based on term appraise the semantic information between document pairs through unsupervised indicators [1], e.g., via TF-IDF vectors [2], BM25 [3], LDA [4]. In querying document, retrieving and searching information, these approaches have been successful [1]. The rule-based method requires experienced experts to summarize the rules [5,6], and the stability of the model depends on the knowledge structure of the experts, and there may be contradictions between the rules given by different experts. In order to overcome the shortcomings of rule-based methods, Bengio *et al.* proposed a document recognition method based on machine learning [7]. The main method of machine learning is to divide documents into multiple categories and then classify them. The classic machine learning classification includes Hidden Markov Model (HMM) [8], Maximum Entropy Model (MEM) [9], Maximum Entropy Markov Model (MEMM) [10], Conditional Random Field (CRF) [11] and Support Vector Machine (SVM) [12] can be used for document recognition. These methods have achieved good results in different fields of the corpus, but in the training process, it is necessary to design features for specific fields first. The effect of the model mainly depends on the selection of features, and the generalization ability is not strong.

In recent years, a variety of deep neural network models for text matching have also been proposed [13,14], which can be recursive or convolutional neural The network layer captures the semantic dependence (especially the order dependence) in natural language. Lample *et al.* proposed a multi-language general's BiLSTM-CRF model that uses word embedding as a feature to identify named entities [13]. Pinherio *et al.* [14] first used CNN combined with CRF to achieve good results in CONLL2003 corpus. Huang *et al.* [15] constructed a BiLSTM-CRF model with artificially designed spelling features, which achieved an F1-measure of 88.83% in CONLL2003 corpus. Chiu and Nichols *et al.* established the BiLSTM-CNNs model to achieve an F1-measure of 91.62% in CONLL2003 corpus [16]. Dernoncourt *et al.* designed an easy-to-use neural network entity recognition tool named NeuroNER [17], which allows users to directly tag entities and perform training

[*] Corresponding author. Zuping Zhang
E-mail address: zpzhang@csu.edu.cn

and prediction by using the web graphical interface. Crichton *et al.* proposed a multi-task learning method for biomedical named entity recognition [18], which increased the average $F1$-measure by 0.8% compared to single-task learning. Shen *et al.* proposed a named entity recognition method based on deep active learning [19], and deep active learning has also achieved great results in the fields of medicine and imaging [20–22], compared to the deep learning method, it requires only a small amount of training data to get the same effect.

However, the existing deep models mainly involve matching sentence pairs, such as paraphrase recognition, answer selection in documents, omitting keywords, entities, or complex interactions between sentences in longer documents. Therefore, although the document is important for matching, it has not been fully studied.

Semantic matching between long documents is largely an untapped area although there are many datasets for sentence matching. However, as far as we know, there is no public dataset of tags for matching long documents. To facilitate the evaluation and further study of the documents, this paper created two labeled datasets, one annotated whether the project feasibility report document pairs (from different projects) belong to the same project, and the other annotated whether the document pairs belong to the same topic. These documents are the scientific and technological projects declared from the subsidiaries of State Grid Hunan Electric Power Co., Ltd. Note that similar to most other natural language matching other natural language matching models, all the methods proposed in this article can also be easily applied to other languages. Specifically, we have made the following contributions:

(1) First, we propose the Concept Graph (CG), which treats a document as a weighted graph of concepts. A keyword or a group of closely connected keywords represents a concept vertex. We use the sentences in the document associated with each concept as a local comparison with the same concept that appears in another document. In addition, we use the weighted edges to connect two conceptual vertices in the document, and use edges to indicate their interaction strength. CG not only captures the essential semantic unit in the document, but also provides a method for anchoring comparison between two documents based on discovered concepts.

(2) Second, we propose a divide and conquer framework to match a pair of documents based on the constructed CG and Graph Convolution Network (GCN). The idea is that for each concept vertex appearing in two documents, we first obtain a local matching vector via a series of text encoding schemes (including neural encoding and term-based encoding). Further, the multi-convolution mixed residual CNN (MCMR-CNN) module is used to obtain local attention information and improve the sensitivity of concept boundary information.

(3) Finally, based on the output of MCMR-CNN as the input, we propose a concept-based Multi-Feature Semantic Fusion Method (MFSFM) , where first design a Contextual Multi-Feature Embedding (CMFE) structure to improve text representation. CMFE performs multi-feature semantic enhancement through multiple features in the dataset, and then performs multi-level feature enhancement through the CNN network. Compared with RNN-based sequential modeling, the MFSFM decomposes the matching process into partial matching sub-problems on the graph. Extensive experiments show that the algorithm we proposed has made significant improvements in matching news pairs. Specifically, the classification accuracy of our proposed MFSFM on the two datasets has been improved by 13.17% and 19.82%.

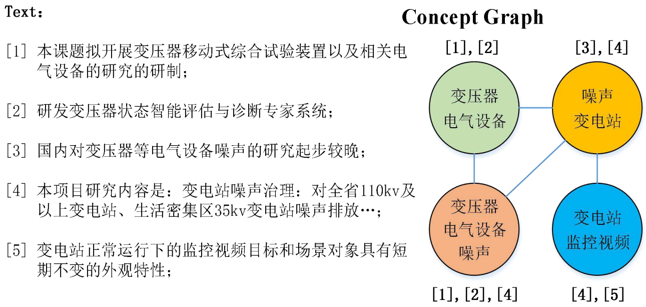
2. Related Works

Traditional document representation methods mainly include vectors such as Bag Of Words (BOW) [23], Term Frequency Inverse Document Frequency (TF-IDF) [2], Latent Dirichlet Allocation (LDA) [4], and compute the distance among the vectors. However, semantic information can not be captured, and generally fail to achieve good performance.

Graphical document representation is proposed in order to better capture the semantic distance. Most of the existing graphical document representations can be summarized into four categories: word, text, concept and hybrid graph. In the word graphs, the words in the document are used as fixed points, and edges are constructed based on syntax analysis [24], co-occurrence [1,25] or the previous relationship [26]. In text graphs, sentences, paragraphs or documents are all utilized as vertices, and use word co-occurrence, location [27], text similarity [28] or hyperlinks among documents [29] to build edges. We link the terms of documents to real-world concepts based on knowledge storehouses such as DBpedia [30] in the concept map, and construct edges based on semantic and syntactic rules. Hybrid graphs [31,32] are composed of vertices and edges, which them are different types.

In recent years, there have emerged different neural network architectures for matching documents pair tasks [18,33,34]. These representation-focused models usually convert the document pairs into a context vector via the Siamese network, and then according to the context vector, use a fully connected layer or scoring function to give a matching result [13,35]. For models that focus on interaction, they extract all the features of paired interactions between words in a documents pair, and combine the interaction information through a Deep Neural Network (DNN) to derive matching results [14]. However, these neural network models do not make full use of the inherent structural characteristics of long text documents. Therefore, these models underperformer in matching long text pairs. There are also some researches using knowledge [34], hierarchical attributes [36] or graph architecture [25] for matching long text. On the contrary, the proposed MFSFM represents the document through a novel graphical notation, and then combines the notations with GCN. Soon after, there have merged pre-training models such as BERT [37], which they can also be used for text matching. However, these models has high complexity and is difficult to meet the speed requirements in practical applications.

The previous GCN architecture was mainly used to make up missing attributes/links [38], classification [39] or node clustering, but they were all within the scope of a single graph such as a knowledge graph, social or citation network. In this paper, the proposed CG uses a simple method to represent project documents through weighted undirected graph, which actually helps to decompose these documents into sentences subsets, each of which focuses on different concept or subtopics. Compared with the previous use of Natural Language Processing (NLP) to deconstruct the document, our method can better reflect the semantics of the document.


Text：

[1] 本课题拟开展变压器移动式综合试验装置以及相关电气设备的研究的研制；

[2] 研发变压器状态智能评估与诊断专家系统；

[3] 国内对变压器等电气设备噪声的研究起步较晚；

[4] 本项目研究内容是：变电站噪声治理：对全省110kv及以上变电站、生活密集区35kv变电站噪声排放…；

[5] 变电站正常运行下的监控视频目标和场景对象具有短期不变的外观特性；

Fig. 1. Concept Graph representation (taking a piece of text as an example).

In addition, the manual review method is not applicable in the case of large amount of data, and the word-based document duplicate checking method can not mine the in-depth semantic information of the document. These models cannot effectively mine the semantic information of long texts. In order to match document pairs more accurately and easily, and considering the location features and part-of-speech features of the key words, this paper proposes a Multi-feature Semantic Fusion Model (MFSFM) to identify citation entities. The model does not require manual rules and templates, and it can also better identify citation entities based on the extracted generic features.

3. Methods

3.1 Concept Graph

As mentioned earlier, the Concept Graph (CG) represents a document as an undirected extended graph. Firstly, the document are decomposed into subsets of sentences, each of which aligns to a different concept. In a document D, we define a graph G_D as a CG. Each of G_D is initially called a concept, and it is a keyword or a group of highly related keywords [1]. The above is also the most common concept mentioned in the sentence. Therefore, the beginning will have its own set of sentences, which are disjoint.

As shown in Fig. 1, it describes how we can transform a document into a Concept Graph. We can use standard keyword extraction algorithms such as TextRank [27] to extract the keywords "变压器", "电气设备", and every other two concepts from the document. In CG, each concept is a subset of closely related keywords. We first group keywords into concepts, and then append each sentence onto its most relevant concept vertex [1]. For example, in Fig. 1, sentence 1 and sentence 2 mainly discuss the relationship between "变压器" and "电气设备" , so it is appended to the concept (transformer, electrical equipment). Therefore, we use a key concept map to denote the original document. Each concept map has a subsets of sentences and the topology relationship between them. Fig. 2 indicates the alignment of the discovered concepts and construction of the CG of the document. Herein, the detailed steps are described for splitting the document and merging the CG:

(1) Constructing KeyGraph: Given a document D, we apply TextRank [27] to extract named entities and keywords. Further, we build a keyword co-occurrence graph based on the set of found keywords, called Key Graph (KG), where each key is a vertex. If two keywords appear in the same sentence at the same time, we will connect them through constructing an edge. To further improve the model, we can implement common citation analysis and synonym analysis to combine keys with the same meaning. Since time complexity, these operations does not work.

(2) Concept detection: The architecture of KG reveals the interaction relationship between keywords. We will build a densely connected subgraph in Key Graph when a subset of keywords are highly correlated, we call it a concept [1]. Further, we use community detection algorithm to extracte concepts. The community detection algorithm can divide KG G_{key} into a group of communities $P = \{P_1, P_2, ..., P_{|P|}\}$, where each community P_i contains a keyword of a certain concept. Each keyword may appear in multiple concepts by using overlapping community detection. Since the number of concepts in different documents varies greatly, we use an algorithm based on the centrality score of betweenness [40] to detect keyword communities in KG. It is worth noting that each keyword is directly utilized as a concept. The advantage of concept detection is that it reduces the number of vertices and increases the matching speed.

(3) Attaching sentence: After discovering concepts through keywords, we further group sentences by concepts by similar methods. Then, the cosine similarity is calculated between sentence and concept. We use TF-IDF vectors to represent them respectively [2]. Each sentence is attached to the concept that is most similar to that sentence. Those sentences that do not include concepts match will be appended to virtual vertices. It does not include any keywords.

(4) Constructing edges: The relationship between concepts is reflected by putting edges between concepts. For each vertex, we express its sentence set as a series of sentences connected to it, and use TF-IDF similarity to calculate the edge weight between the two vertices. Note that, we can use other ways to determine the edge weight, but constructing an edge through TF-IDF is better because it will generate a CG, which is more closely connected.

As shown in Fig. 2(a), we use the above steps to address a pair of documents D_A and D_B while performing item matching. It is different that for each common concept vertices, we align the CGs of the two documents according to the concept vertices, and for local comparison, we merge the sentence sets in D_A and D_B.

3.2 Document Pair Matching

Given the merger of the two documents D_A and D_B introduced, a pair of documents are matched by matching the sentence sets. As shown in Fig. 2, we match the set of sentences in D_A and D_B related to each concept. Then, we use multiple graph convolutional layers to aggregate the local matching results into the final result, and use a "divide and conquer" manner to match a pair of words [1]. To overcomes the disadvantages of previous algorithms and capture more semantic interations in longer texts, we use a graphics perspective to spread the text representation from a grid perspective.

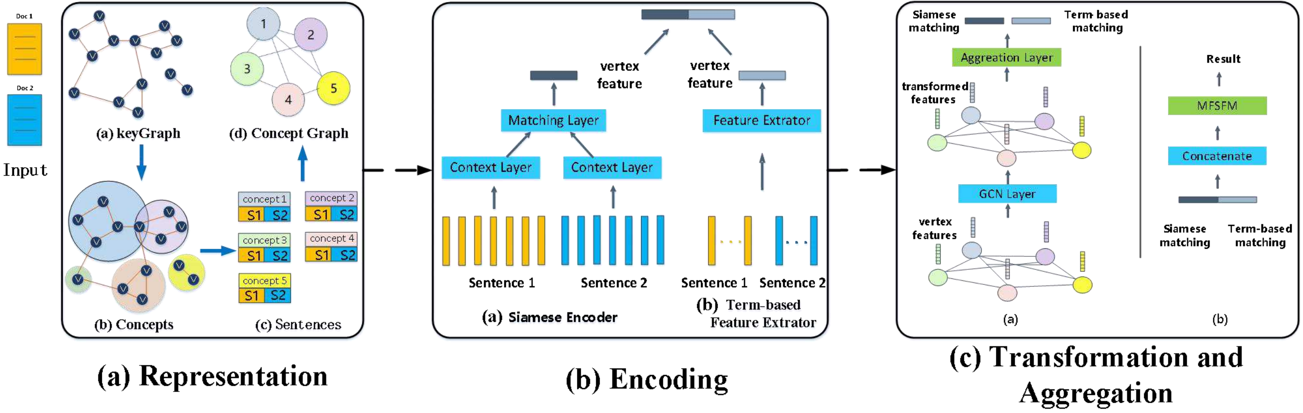


Fig. 2. The outline of the way we construct a CG from documents pairs and classify it through a GCN (Similar to Ref. [1]).

As shown in Fig. 2, it presents the overall architecture of the proposed MFSFM, including four steps: a) expressing documents pairs through a single merged CG, b) studying the multi-viewed matching feature from each concept vertex, c) structuring transformation of the local matching graph by the features of the convolutional layer, and d) grouping local matching features to obtain the final result. The above four steps can be trained in end-to-end manner.

Given the grouped CG G_{AB}, MFSFM first learn a fixed-length matching vector for each concept $v \in G_{AB}$ to represent the TF-IDF semantic similarity between $C_A(v)$ and $C_B(v)$ and the sentence sets from recording D_A and D_B separately. It means that the two documents matching will be converted to match sentence sets pair for each vertex. Especially, local matching vectors are generated according to term-based techniques and neural networks. Siam network encoder [41] is applied to each vertex $v \in G_{AB}$ to transform the word embedding of $\{C_A(v), C_B(v)\}$ into a hidden feature vector $m_{AB}(v)$, which is fixed-sized.

In this paper, the Siamese structure is used to take $C_A(v)$ and $C_B(v)$as inputs. Then, $C_A(v)$ and $C_B(v)$ are encoded into two context vectors by the context layers. This can achieve the purpose of sharing the same weights in Fig. 2(b). In the context layer, one or multiple BiLSTM or CNN layers are included. The purpose of BiLSTM and CNN is to capture the contextual message in $C_A(v)$ and $C_B(v)$. Define $t_A(v)$ and $t_B(v)$ as the context vectors, which are used to obtained for $C_A(v)$ and $C_B(v)$, respectively. Then, we calculate $\mathrm{m}_{\mathrm{AB}}(v)$ for v through the subsequent aggregation layer [1]. $\mathrm{m}_{\mathrm{AB}}(v)$ concatenates the element-wise multiplication and the element-wise absolute difference of the context vectors A and B, i.e.,

$$\mathrm{m}_{AB}(v) = (t_A(v) \circ t_B(v), |t_A(v) - t_B(v)|), \quad (1)$$

where ∘ represents Hadamard product [1].

According to different similarity algorithm, there are different calculation method for matching vertors. There are usually 4 indicators (TF-IDF cosine similarity, TF cosine similarity, BM25 cosine similarity and Jaccard similarity of 1-gram) to calculate the term-based similarity between $C_A(v)$ and $C_B(v)$. As shown in Fig. 2(b), in this paper, we use the four similarity scores to concatenate the comprehensive similarity into another matching vector $\mathrm{m}'_{AB}(v)$ of v. It is different from Ref. [1]. Matching aggregation through GCN need to aggregate the local matching vector into the final matching score of documents pairs. In Ref. [38], the function of the GCN filter is recommended to obtain the patterns shown in CG G_{AB} on multiple scales. Generally, a graph $G = (V, E)$ is considered as the input of GCN, N vertices $v_i \in \mathrm{V}$ and $e_{ij} = (v_i, v_j) \in E$. In addition, the vertex feature matrixs represented by $\mathrm{F} = \{\mathrm{f}_i\}_{i=1}^{N}$ are included in the input. For vertex v_i, f_i is the feature vector. Then, CG G_{AB} of documents pairs D_A and D_B, which contains the connected matching vector on each vertex into GCN, so that f_i of v_i in GCN is expressed as:

$$\mathrm{f}_i = \left(m_{AB}(v_i), \mathrm{m}'_{AB}(v_i)\right). \quad (2)$$

Next, we slightly bewrite the GCN layer used in Fig. 2(c) [38]. The weighted adjacency matrix of G_{AB} is given by $\mathrm{A} \in \mathbb{R}^{N \times N}$ where $A_{ij} = w_{ij}$ that is the TF-IDF similarity between vertex i and j. Denote B as a diagonal matrix, and let $B_{ij} = \sum_j A_{ij}$. The input layer of GCN is $H^{(0)} = \mathrm{X}$. The original vertex features is contained by $H^{(0)}$. We express $H^{(l)} \in \mathbb{R}^{N \times M_l}$ as the hidden representation matrix in the l^{th} layer. Then, the following graph convolution filter was applied to the previous hidden representation by each GCN layer:

$$H^{(l+1)} = \sigma(\tilde{B}^{-\frac{1}{2}}\tilde{A}\tilde{B}^{-\frac{1}{2}}H^{(l)}T^{(l)}), \quad (3)$$

where $\widetilde{A} = A + E_N$, E_N is the identity matrix, $\widetilde{B}$ is the diagonal matrix, and its value is $\widetilde{B}_{ii} = \sum_j \widetilde{A}_{ij}$. $\widetilde{B}_{ii}$ is the adjacency matrix of the graph G. $\widetilde{A}_{ij}$ is the degree matrix.

The trainable weight matrix is indicated as $T^{(l)}$ in the l^{th} layer. $\sigma(\cdot)$ means an activation function (Sigmoid or ReLU function, etc.). On the graph G_{AB}, the first-order approximation of the local spectral filter inspires this graph convolution rule [38]. The interaction pattern between vertices can be extracted when employed recursively [1]. Finally, according to the obtained average value of the hidden vectors of all vertices in the last layer, we merge the hidden meaning in the final GCN layer into a single vector of fixed length. We can employ a classifier, such as Multi-Layer Perceptron, to compute the final matching score based on m_{AB}.

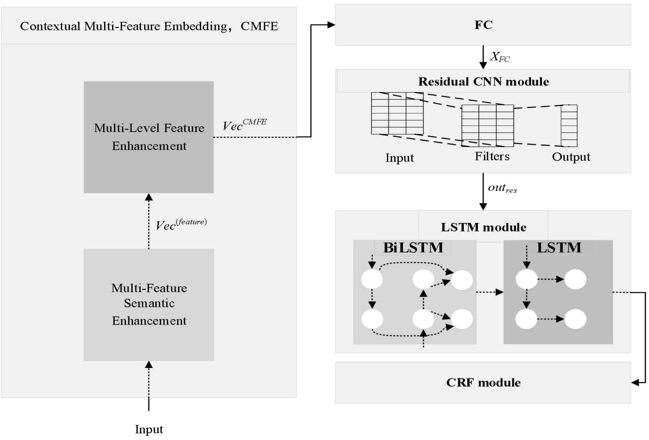


Fig. 3. Multi-Feature Semantic Fusion Model (MFSFM) architecture.

Apart from the above matching vector m_{AB}, other global matching features are appended to the m_{AB}, which further expand the feature set. We encode two documents to calculate these additional global functions, where we use the latest language model such as BERT [37] as encoder. In addition, we also can calculate the term-based similarityies as the global features.

3.3 Model construction

The MFSFM's architecture is shown in Fig. 3. According to the citation dataset constructed in Sec. 3, MFSFM first design a Contextual Multi-Feature Embedding (CMFE) structure to obtain word vectors to better express semantic information, and use the designed residual CNN module to obtain entity boundary information of variable length, design LSTM module to further obtain context information and clarify timing, and finally use CRF module to perform Entity recognition. Secondly, considering the uncertainty of the entity boundaries in the citations divided by division granularity, for example, each author in Author list entity is generally composed of 2 to 3 divisions, but Title entity may be composed of 4 to 20 divisions (According to the division granularity of Chinese and English citations, the authors in Chinese citations consist of 2 to 3 characters, and the authors in English citations consist of 2 to 3 words. Title entity is similar.), so MFSFM constructed a multi-convolution kernels mixed residual CNN module to obtain the local attention and entity boundary information. Thirdly, MFSFM used a LSTM module which composed with BiLSTM and one-way LSTM to enhance the timing information learning. Finally, MFSFM used the CRF module to identify the citation entity.

The citation entity recognition first needs to generate the text representation for words or characters of the citations according to the division granularity, mainly including one-hot representation and distributed representation [42]. Because the one-hot representation does not take into acco unt the relevance of the words and may present a "highdimensional disaster", we choose distributed representation. Existing distributed representation models include neuro-probabilistic language models [43], word2vec [44], BERT [37], XLNet [45] and so on.

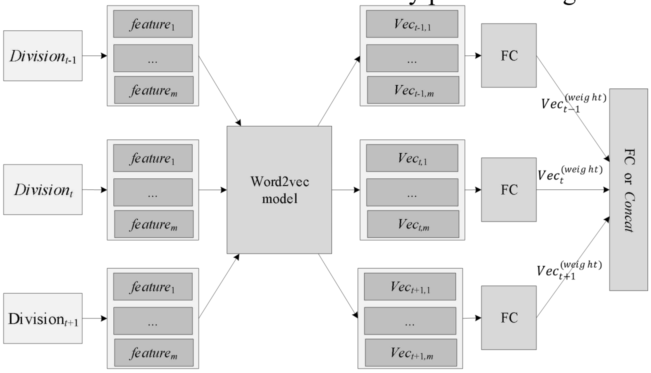


Fig. 4. Multi-Feature Semantic Enhancement (an example of n=3).

Word2vec can represent each word as a low-dimensional vector to compress the data scale, which can capture less contextual information. And it is small scale, fast and easier to learn, so this paper used word2vec to get a preliminary text representation. As for the citation dataset in Sec. 2 not only having the characters (words) features, but also having part-of-speech features and relative position features, this paper proposed CMFE method. The CMFE mainly includes two processes: Multi-Feature Semantic Enhancement (Fig.3) and Multi-Level Feature Enhancement (Fig. 4). The main steps are as follows:

Multi-Feature Semantic Enhancement steps:

i) The word vector matrix $W^{feature_i}, i = 1,2,...,m$ (m represents the number of feature) of each feature in dataset is obtained by using the word2vec model.

ii) For each feature in each division, input it to the matrix $W^{feature_i}, i = 1,2,...,m$ and getting the corresponding feature vector. Then, using the fully connected (FC) layer to obtain the weighted word vector (without the bias vector), FC can be trained and can reflect the semantic influence of different feature.

Noting that the weighted word vector corresponding to the current division t is $Vec_t^{(weight)}$, and each feature vector of the current division t is $Vec_{t,i}, i = 1,2,...,m$, the weight of FC for division t is $weight_{t,i}, i = 1,2,...,m$, then $Vec_t^{(weight)}$ can calculate by (4) ($\odot$ is element-wise multiplication operation).

$$Vec_t^{(weight)} = \sum_{i=1}^{m} weight_{t,i} \odot Vec_{t,i} \qquad (4)$$

iii) The window parameter n is used to obtain the context information that needs to be included in the final multi-feature semantic enhancement vector to reflect the semantic. It mainly uses FC (without bias vector) calculation or concatenate the n divisions weighted word vector. The n divisions are split into follows: front has n-(n-1)/2 divisions, rear has (n-1)/2 divisions, and current division (the division is not smaller than 1, and not bigger than the max division number).

Noting the max division number is T, current division is t, the weight of FC for n divisions is $weight_{t+i}, 0 < t + i \le T, i = \left(-\left(n - \left\lfloor\frac{n-1}{2}\right\rfloor\right), \ldots, \left\lfloor\frac{n-1}{2}\right\rfloor\right)$, so $Vec_t^{(feature)}$ (the multi-feature semantic enhancement vector) can calculate by (5) (using FC, $\odot$ is element-wise multiplication operation), (6) (using concatenation, $\sum\oplus$ is cumulative concatenation operation), in (5) and (6), $0 < t + i \le T$.

$$Vec_t^{(feature)} = \sum_{i=-\left(n-\left\lfloor\frac{n-1}{2}\right\rfloor\right)}^{\left\lfloor\frac{n-1}{2}\right\rfloor} weight_{t+i} \times Vec_{t+i}^{(weight)} \qquad (5)$$

$$Vec_t^{(feature)} = \sum \oplus_{i=-\left(n-\left\lfloor\frac{n-1}{2}\right\rfloor\right)}^{\left\lfloor\frac{n-1}{2}\right\rfloor} Vec_{t+i}^{(weight)} \qquad (6)$$

Multi-Level Feature Enhancement steps:

Considering that multi-feature semantic enhancement only extracts shallow features, and it does not specifically capture deep features (relevant information) between data divisions. In order to express the semantics of data with a matrix of word vectors, that is, to better express the different semantics of a word between different data. And the convolution operation in the CNN network can obtain relevant information of data division by expanding the receptive field. Multi-level feature enhancement uses a two-layer CNN network (using one-dimensional convolution) to obtain the two-level feature vectors. The two-level feature vectors are combined with multi-feature semantic enhancement vector to get the CMFE vector.

In Fig. 5, $Vec = [Vec_1, Vec_2, Vec_3, \ldots, Vec_T]$ is the multi-feature semantic enhancement's output, T is the max division number, and the one-dimensional convolution's outputs $h_t^{(1)}$ and $h_t^{(2)}$ are the two-level feature vectors, so CMFE vector Vec_t^{CMFE} is shown in (7) ($\oplus$ is concatenation operation).

$$Vec_t^{CMFE} = Vec_t \oplus h_t^{(1)} \oplus h_t^{(2)} \qquad (7)$$

The CMFE method can be used to obtain the word vector representation of each division data, and the multi-feature is used to strengthen the semantic information and the simple CNN network to strengthen the hierarchical information, making the subsequent learning process easier.

4. Results

We evaluate the proposed method to identify whether a pair of feasibility study project reports belong to the same project (or event), and whether they have the same theme. In fact, the proposed matching scheme for document pairs has been deployed in the project declaration application for project verification. Please note that traditional project document review methods include manual verification and character-based duplicate checking methods. Although manual verification has a high accuracy rate, it requires a lot of human resources; the character-based duplicate check method only judges the repetition rate at the character level, and cannot infer whether the document pair belongs to the same project or the same topic at the semantic level. Therefore, manual verification methods and character-based duplicate verification methods are not available here. It is not even possible to determine how many project clusters exist. This is different from news document pair matching. There are uncertainties about the number of project categories. The topic of the project document is fixed. The task of classifying whether two project declaration documents belong to the same project or the same subject is crucial.

In our tasks, "project" refers to a task set up to solve a certain problem. Multiple tasks may publish documents with different narratives and wordings in the project. Note that our goal is different from traditional event references [46] or SemEval-2018 Task 5[47]. Their task is to detect all events mentioned in the document (or actually "actions", such as shooting, car accidents) [1]. In contrast, although a project document may mention multiple entities or even domain-specific terms, the "project" in our data set always refers to the problem that the document intends to study. Our task is to determine whether two documents intend to study the same topic.

We tested the following benchmarks:

(1) Based on DNN models: ARC-I [33], ARC-II [33], DSSM [48], CDSSM [15], DUET [49] and MatchPyramid [14]. To evaluate these models, we employ the implementation of MatchZoo [50].

(2) Similarity based on terminology: BM25 [3], LDA [4] and SimNet (the above mentioned four text pair similarities are being extracted through multiple classification).

(3) Based on the large-scale pre-trained language model BERT [37]. The basic idea is that the bi-transformer is responsible for extracting features, and then the entire network adds a fully connected linear layer as fine-tuning.

In this paper, we focus on how to better match long text. Therefore, in our method or baseline, any short text information (such as headings) have been abondoned. Pratically, the interaction of two projects is not limited to "whether they belong to the same project". The proposed MFSFM can identify general interaction between projects, for example, whether there are two transformer feasibility reports describing transformer noise. We use the labeled training data to define and supervise the interaction. The interaction contains the same project or the same topic. We can not assume the feasibility of other information (such as titles) for these experiments. Table 2 evaluate different variants of the proposed MFSFM to show the impact of different sub-modules. In the model name. In Fig. 2, "Siam" means an encoder using a Siamese, and "Sim" means an encoder using a term-based similarity. "CG" means that in Concept Graph (CG), if community detection is not used, keywords are directly used as concepts, and "CG_{cd}" represents that each concept vertex in CG. These vertex includes a keywords set grouped by community detection. Therefore, for each vertex, a matching vertor is produced; "GCN" indicates that we take the vertices vector of the GCN layer to convolve the local matching. Finally, "$BERT_g$" means using other global functions provided by BERT, and "SIM_g" means using the above four term-based similarity measures. These features are appended to the matching vector m_{AB} of the graph merge for final classification [1].

4.1 Datasets

For matching long document tasks, as far as we know, there is no public available data set that can be used. In this paper, we constructed two datasets: the Chinese Feasibility Study Same Project Data Set (CNSR) and the Chinese Feasibility Study Same Subject Data Set (CNSI). These datasets are all marked by professional editors. They contain long-form feasibility technology documents collected from China's Hunan State Grid Electric Power Co., Ltd., covering various topics in various areas of the company. The CNSR data set contains 4678 pairs of feasibility study reports with tags. These tags indicate whether a pair of project documents are describing projects in the same field. Similarly, the CNSI data set contains 2464 pairs of tagged documents, indicating whether the two projects belong to the same subject. The average number of words in all documents in the data set is 9034, and the maximum is 32461.

In these data set, we only marked the main research items of the feasibility study report. Please note that we do not generate randomly the negative samples in the two datasets. Rather than, we choose project document pairs that include similar items (keywords), and exclude samples whose TF-IDF similarity is below a certain threshold. Table 1 shows the detailed classification of these two datasets.

Table 1: Description of evaluation datasets.

Datasets	Pos Samples	Neg Samples	Train	Dev	Test
CNSR	2010	2668	3275	702	701
CNSI	1200	1264	1725	369	370

For these two data sets, we use 70% of all samples as the training set, 15% as the validation set, and the remaining 15% as the test set. In this paper, we need to ensure that the different segmentation do not include any overlap, which avoids data leakage. The indicators used for performance evaluation are the accuracy of the binary classification results and the $F1$ score. For each evaluation method, we take training for 10 periods, and then select the period with the best verification results for evaluation on the test set.

4.2 Experimental setting

We use Stanford CoreNLP for word segmentation (Chinese text) and named entity recognition. For the concept interaction graph construction with community detection, we set the minimum community size (the number of keywords contained in the concept vertices) to 2, and the maximum size to 6.

In our neural network model, there are word embedding layers, Siamese encoder, graph convolution layer and classification layer. In word embedding layers, the pre-trained word vectors are loaded and repaired during the training process. The embedding of words outside the vocabulary is set to a zero vector. In the Siames enocder, we employ 1-dimensional convolution and 64 filters, followed by the ReLU and the Max Pooling operation. In graph convolution, we use 3-layer GCN [38] to conduct experiments on the CNSS dataset, and use 3-layer GCN to conduct experiments on the CNSE dataset. The output size of the GCN layer is set to 32 when the vertex encoder has a 4-dimensional feature; The output size of the GCN layer is set to 128 when the vertex encoder is a Siamese encoder. Note that, except for the last layer. In the GCN layer, we always set the output size to 32. In the last classification layers, there are a linear layer with an output size of 32, a ReLU layer. It is worth nothing that this classifier is also used for the benchmark SimNet. We use tensorflow 2.0 to implement the proposed MFSFM. The experiment without BERT was performed on a MacBook Pro equipped with a 2 GHz Intel Core i7 processor and 8 GB of memory. L2 weight attenuation is used for all trainable variables, parameter $\lambda = 2$ e-16. The loss rate between every two layers is 0.005. The gradient clipping with a maximum gradient norm of 5:0 is used in this paper, and the ADAM optimizer [51] is also applyed, where $\beta_1 = 0.85, \beta_1 = 0.99$ $, \epsilon = 1e8$. The learning rate warm-up scheme to increase its inverse exponent is set from 0.0 to 0.001 in the first 1500 steps, and then keep a constant learning rate in the rest of the training. The maximum number of training epochs is set to 20 in all experiments.

4.3 Analysis

In order to verify the effectiveness of the Contextual Multi-Feature Embedding (CMFE) proposed in this paper, the BiLSTM-CRF model proposed in [16] was used as the citation entity recognition model, and the CMFE was compared with CBOW and Skip-gram. Among them, the parameters of the CBOW and Skip-gram algorithms were set as follows: the context window=5, the number of negative samples=10, word2vec size=128. The parameters of the CMFE were set to use CBOW and Skip-gram algorithms (using the same parameters as before), and word2vec size = 128 (64 dimensions for multi-feature semantic enhancement (the parameter n used for multi-feature semantic enhancement was 3). 64 dimensions for Multi-level features enhancement (the size of the convolution kernel used was 3). It can be observed in the table that the model using the CMFE is significantly higher than other methods in entity recognition. And CMFE (Skip-gram) obtained the best recognition effect, the

BiLSTM-CRF model can obtain an average $F1$-measure of 88.80% on the Chinese citation dataset, and an average $F1$-measure of 88.84% on the Chinese-English mixed citation dataset. Compared with the original CBOW and Skip-gram methods, the average $F1$-measure is increased by more than 10%.

The performance of all comparison methods on the two datasets can be concluded in Table 2. Note that, the idea of vectorizing documents through concept graphs comes from Ref.[1]. Based on this, this paper proposes the MFSFM model, which uses the result of document vectorization as input through the concept graphs. The proposed MFSFM achieves the best performance on both datasets and is significantly better than all other methods, which is caused by two reasons. First, the two documents are aligned along the corresponding semantic unit to facilitate conceptual comparison because the input of the document pair is reorganized into a CG. Second, the proposed MFSFM encodes the local comparisons around different semantic units into local matching vectors, and aggregates them through graph convolution, taking into account the semantic topology. Therefore, it solves the problem of matching documents through divide and conquer, and is suitable for processing long texts.

Table 2. Text representation comparison results

Method	Entity	Precision(P)	Recall(R)	$F1$-measure	Avg $F1$-measure
CBOW	Author list	82.57	78.46	80.46	72.81
	Title	60.50	67.25	63.70	
	Publisher	61.24	70.01	65.33	
	Time	83.33	80.24	81.76	
Skip-gram	Author list	86.10	86.50	86.30	77.31
	Title	61.39	70.49	65.67	
	Publisher	65.49	72.14	68.65	
	Time	88.73	88.47	88.60	
CMFE(CBOW)	Author list	93.84	93.55	93.69	87.00
	Title	76.30	79.83	78.03	
	Publisher	81.10	84.24	82.64	
	Time	95.79	91.55	93.62	
CMFE(Skip-gram)	Author list	95.84	94.61	96.12	88.80
	Title	80.53	81.95	81.23	
	Publisher	85.01	87.69	86.33	
	Time	92.97	90.14	91.53	

The effect of Graphical Decomposition: By comparing method No.11 in Table 3 with method No.6. They with same word vector use Neural Networks (NN) for encoding text. The pivotal difference is that No.11 compares documents pair on CG in a vertex-by-vertex decomposition in our methods. It can be observed that the performance of algorithm No.11 is outperformer algorithm No.6. Equally, comparing our algorithm No.14 with algorithm No.9, both of which apply the same term-based similarity. However, our method MFSFM greatly outperformer No.9 via using graphical decomposition. Thus, it can be concluded that graph decomposition can significantly improve matching performance for long text. It is worth nothing that the No.6 algorithm lead to poor performance. This is because they are deep text matching algorithm. Besides, they are mainly invented for matching sequence and can not obtain meaningful semantic information in project document pairs at all. It is difficult for matching document pairs to obtain a suitable context representation when the context is too long. For NN models that focus on interaction, most interactions among words is meaningless for two long documents.

The effect of graph convolution: In our comparative experiments, we take comparative experiment of algorithm No.12 and algorithm No.11, and algorithm No.15 and algorithm No.14. it can be observed that the performance improves significantly in the two datasets by merging GCN layers. Each vertex hidden vectors are updated by each GCN layer integrate its neighboring vertices into vectors. Therefore, local matching features needs to be studied how to aggregate them into the final result graphically in the GCN layer. We take comparative experiment comparing algorithm No.13 and algorithm No.12, and algorithm No.16 and algorithm No.15, it can be saw that the community detection will bring about briefly worse performance because the conceptual vertices that directly use keywords can offer more anchor points to compare documents pairs. As mentioned earlier, the community detection technology refers to a group of keywords forming a concept instead of a keyword. However, consistent keywords can be highly grouped together by community detection, and the average size of CG can be reduced from 35 vertices to 16. The total training time of the proposed MFSFM can be reduced by 53.6%, and the same is true for test time. Therefore, you can choose whether to use community detection to weigh accuracy in exchange for acceleration.

Time complexity: For the keywords of technical project document, in real-world science and technology project declaration system, we usually extract them through efficient tools and predefined vocabulary rules. Below we will explain the time complexity of the proposed MFSFM through the process of constructing CG. In two documents datasets, denote n as the number of sentences, m as the number of unique words, and q means the number of unique keywords. The operation of community detection needs $\mathcal{O}(q^3)$, and constructing a keyword map requires $\mathcal{O}(nm+q^2)$. The operation of attaching sentences and calculating weight needs $\mathcal{O}(nm+m^2)$ complexity. For the final step, that is results classification, due to the proposed MFSFM is not big and can effectively address document pairs, the complexity of the classification operation can be ignored.

The effect of multi-view matching: In our comparative experiments, we take comparative experiment of algorithm No.17 and algorithm No.15. It can be observed that the concatenation vectors (from different view matching, such as Siamese encode features and term-based) can further outper former other algorithms, which proves the benefit of concatenating multi-view matching vectors. We also take comparative experiment of algorithm No18, No.19, No.20 with algorithm No.17, it can be concluded that the more global features always underperformer other algorithm. These global features includes documents pairs similarities and/or encoding. It indicates that the main factors of improving performance are decomposition and convolution of graph. This is similar to Ref. [1]. This is because the models have learned to summarize that local comparisons should be putted into global semantic relations, and the additional design of global features is of no avail.

Model size and parameter sensitivity: In our experiments, the largest model without BERT is No.18, which only includes about 54K parameters. In contrast, there are 130M-340M parameters in BERT. However, the proposed MFSFM is greatly better than BERT. In addition, we conduct some tests in the model about the sensitivity of different parameters. It can be found that the performance of the 2-3 GCN layers is better. Furthermore, adding more GCN layers will not be better than the 2-3 layers, but if it is zero or only one GCN layer, the performance will be worse. In addition, hidden vectors with sizes between 32 and 256 have good performance in GCN. And larger size will not cause a significant improvement in performance. When we construct CG, we need to select the size of the community for the optional community detection step. It can be found from experiments that the performance will be worse if the maximum size is from 8 to 10 and the minimum size is from 2~3. This shows that the proposed MFSFM is steady and insensitive compared to the parameters. All in all, the MFSFM proposed in this paper based on concept maps is better than other algorithms.

Table 3. Comparison of accuracy and $F1$-score under different methods based on CNSR and CNSI datasets.

Baselines	CNSR		CNSI		X-MFSFM	CNSR		CNSI	
	Acc	F1	Acc	F1		Acc	F1	Acc	F1
1.ARC-I	61.35	47.61	49.23	65.56	11.CG-Siam	73.42	72.72	74.51	77.42
2.ARC-II	53.25	35.58	51.67	52.94	12.CG-Siam-GCN	**73.63**	**72.45**	**77.93**	**79.68**
3.DUET	54.69	52.77	52.61	59.88	13.CGcd-Siam-GCN	72.23	72.07	75.19	75.97
4.DSSM	57.21	63.81	61.13	69.56	14.CG-Siam	71.66	70.99	74.13	76.49
5.C-DSSM	59.23	47.63	51.57	55.47	15.CG-Siam-GCN	**82.34**	**79.56**	**86.06**	**86.35**
6.MatchPyramid	65.22	53.12	61.25	63.21	16.CGcd-Siam-GCN	80.26	77.33	85.58	86.01
7.BM25	68.31	65.72	65.31	69.72	17.CG-Siam & Siam-GCN	83.56	**81.71**	88.55	89.16
8.LAD	68.64	62.56	61.64	69.56	18.CG-Siam & Siam-GCN-Simgg	83.83	81.52	**89.12**	**89.38**
9.SimNet	70.16	68.29	70.16	73.29	19.CG-Siam & Siam-GCN-BERTg	**83.33**	81.49	88.84	88.79
10.BERT fine-tuning	80.09	78.57	85.09	86.57	20.CG-Siam&Siam-GCN-Siamg&BERTg	83.11	81.47	88.19	88.36

5. Conclusion

This article studies document pair matching. First, we propose a concept map, which represents a document as a weighted map of the concept; second, we propose a divide-and-conquer framework based on the constructed CG and graph convolutional network to match a pair of documents. Finally, we propose a multi-feature semantic fusion model called MFSFM. Compared with sequential modeling based on RNN, MFSFM decomposes the matching process into partial matching sub-problems on the graph. In addition, with the help of professional editors, we created two new datasets for matching long documents, which contained 7100, which we conducted extensive evaluations. Experimental results show that the proposed MFSFM is significantly better than a wide range of latest solutions, including terminology-based and deep learning model-based text matching algorithms.

However, the expressive power of document matching lies in the understanding of the document. Although this paper expresses the document in the form of a concept map, the construction of the concept map is based on document keywords, and its accuracy depends on the semantic understanding of the document. In future research work, we will focus on researching new model frameworks to improve text semantic understanding.

Acknowledgments

This research was funded by the Ministry of Education, China-111 Project (Grant No. B18059)the Hunan Key Laboratory for Internet of Things in Electricity (Grant No.2019TP1016) and Science and Technology Project of SGCC (Grand No. 5700-202158177A-0-0-00).

References

[1] B. Liu, D. Niu, H. Wei, J. Lin, Y. He, K. Lai, Y. Xu, Matching article pairs with graphical decomposition and convolutions, ArXiv Preprint ArXiv:1802.07459. (2018).

[2] S. Robertson, Understanding inverse document frequency: on theoretical arguments for IDF, Journal of Documentation. (2004).

[3] S. Robertson, H. Zaragoza, The probabilistic relevance framework: BM25 and beyond, Now Publishers Inc, 2009.

[4] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, The Journal of Machine Learning Research. 3 (2003) 993–1022.

[5] W.J. Black, F. Rinaldi, D. Mowatt, FACILE: Description of the NE System Used for MUC-7, in: Seventh Message Understanding Conference (MUC-7): Proceedings of a

Conference Held in Fairfax, Virginia, April 29-May 1, 1998, 1998.

[6] A. Borthwick, J. Sterling, E. Agichtein, R. Grishman, NYU: Description of the MENE named entity system as used in MUC-7, in: Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29-May 1, 1998, 1998.

[7] Y. Bengio, Learning deep architectures for AI, Now Publishers Inc, 2009.

[8] D.M. Bikel, R. Schwartz, R.M. Weischedel, An algorithm that learns what's in a name, Machine Learning. 34 (1999) 211–231.

[9] O. Bender, F.J. Och, H. Ney, Maximum entropy models for named entity recognition, in: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, 2003: pp. 148–151.

[10] A. McCallum, D. Freitag, F.C. Pereira, Maximum entropy Markov models for information extraction and segmentation., in: Icml, 2000: pp. 591–598.

[11] S.-H. Na, H. Kim, J. Min, K. Kim, Improving LSTM CRFs using character-based compositions for Korean named entity recognition, Computer Speech & Language. 54 (2019) 106–121.

[12] S. Suthaharan, Support vector machine, in: Machine Learning Models and Algorithms for Big Data Classification, Springer, 2016: pp. 207–235.

[13] S. Wan, Y. Lan, J. Guo, J. Xu, L. Pang, X. Cheng, A deep architecture for semantic matching with multiple positional sentence representations, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016.

[14] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, X. Cheng, Text matching as image recognition, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016.

[15] Z. Huang, W. Xu, K. Yu, Bidirectional LSTM-CRF models for sequence tagging, ArXiv Preprint ArXiv:1508.01991. (2015).

[16] J.P. Chiu, E. Nichols, Named entity recognition with bidirectional LSTM-CNNs, Transactions of the Association for Computational Linguistics. 4 (2016) 357–370.

[17] F. Dernoncourt, J.Y. Lee, P. Szolovits, NeuroNER: an easy-to-use program for named-entity recognition based on neural networks, ArXiv Preprint ArXiv:1705.05487. (2017).

[18] G. Crichton, S. Pyysalo, B. Chiu, A. Korhonen, A neural network multi-task learning approach to biomedical named entity recognition, BMC Bioinformatics. 18 (2017) 1–14.

[19] Y. Shen, H. Yun, Z.C. Lipton, Y. Kronrod, A. Anandkumar, Deep active learning for named entity recognition, ArXiv Preprint ArXiv:1707.05928. (2017).

[20] X.U. Jia, Y. Chunqi, Active deep learning based polarimetric SAR image classification, Remote Sensing for Land & Resources. 30 (n.d.) 72–77.

[21] Z. Liu, Z. Wang, Y. Yao, L. Zhang, L. Shao, Deep active learning with contaminated tags for image aesthetics assessment, IEEE Transactions on Image Processing. (2018).

[22] A. Smailagic, P. Costa, H.Y. Noh, D. Walawalkar, K. Khandelwal, A. Galdran, M. Mirshekari, J. Fagert, S. Xu, P. Zhang, Medal: Accurate and robust deep active learning for medical image analysis, in: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2018: pp. 481–488.

[23] H.M. Wallach, Topic modeling: beyond bag-of-words, in: Proceedings of the 23rd International Conference on Machine Learning, 2006: pp. 977–984.

[24] J. Leskovec, M. Grobelnik, N. Milic-Frayling, Learning sub-structures of document semantic graphs for document summarization, in: LinkKDD Workshop, 2004: pp. 133–138.

[25] G. Nikolentzos, P. Meladianos, F. Rousseau, Y. Stavrakas, M. Vazirgiannis, Shortest-path graph kernels for document similarity, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017: pp. 1890–1900.

[26] A. Schenker, M. Last, H. Bunke, A. Kandel, Clustering of web documents using a graph model, in: Web Document Analysis: Challenges and Opportunities, World Scientific, 2003: pp. 3–18.

[27] R. Mihalcea, P. Tarau, Textrank: Bringing order into text, in: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004: pp. 404–411.

[28] J.W.G. Putra, T. Tokunaga, Evaluating text coherence based on semantic similarity graph, in: Proceedings of TextGraphs-11: The Workshop on Graph-Based Methods for Natural Language Processing, 2017: pp. 76–85.

[29] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank citation ranking: Bringing order to the web., Stanford InfoLab, 1999.

[30] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, in: The Semantic Web, Springer, 2007: pp. 722–735.

[31] B. Rink, C.A. Bejan, S.M. Harabagiu, Learning Textual Graph Patterns to Detect Causal Event Relations., in: FLAIRS Conference, 2010.

[32] C.F. Baker, M. Ellsworth, Graph methods for multilingual framenets, in: Proceedings of TextGraphs-11: The Workshop on Graph-Based Methods for Natural Language Processing, 2017: pp. 45–50.

[33] B. Hu, Z. Lu, H. Li, Q. Chen, Convolutional neural network architectures for matching natural language sentences, ArXiv Preprint ArXiv:1503.03244. (2015).

[34] Y. Wu, W. Wu, C. Xu, Z. Li, Knowledge enhanced hybrid neural network for text matching, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2018.

[35] J. Mueller, A. Thyagarajan, Siamese recurrent architectures for learning sentence similarity, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016.

[36] J.-Y. Jiang, M. Zhang, C. Li, M. Bendersky, N. Golbandi, M. Najork, Semantic text matching for long-form documents, in: The World Wide Web Conference, 2019: pp. 795–806.

[37] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, ArXiv Preprint ArXiv:1810.04805. (2018).

[38] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, ArXiv Preprint ArXiv:1609.02907. (2016).

[39] W.L. Hamilton, R. Ying, J. Leskovec, Representation learning on graphs: Methods and applications, ArXiv Preprint ArXiv:1709.05584. (2017).

[40] H. Sayyadi, L. Raschid, A graph analytical approach for topic detection, ACM Transactions on Internet Technology (TOIT). 13 (2013) 1–23.

[41] P. Neculoiu, M. Versteegh, M. Rotaru, Learning text similarity with siamese recurrent networks, in: Proceedings of the 1st Workshop on Representation Learning for NLP, 2016: pp. 148–157.

[42] J. Turian, L. Ratinov, Y. Bengio, Word representations: a simple and general method for semi-supervised learning, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 2010: pp. 384–394.

[43] A. Mnih, Y.W. Teh, A fast and simple algorithm for training neural probabilistic language models, ArXiv Preprint ArXiv:1206.6426. (2012).

[44] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, ArXiv Preprint ArXiv:1301.3781. (2013).

[45] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, ArXiv Preprint ArXiv:1906.08237. (2019).

[46] C.A. Bejan, S. Harabagiu, Unsupervised event coreference resolution with rich linguistic features, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 2010: pp. 1412–1422.

[47] M. Postma, F. Ilievski, P. Vossen, Semeval-2018 task 5: Counting events and participants in the long tail, in: Proceedings of The 12th International Workshop on Semantic Evaluation, 2018: pp. 70–80.

[48] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using clickthrough data, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, 2013: pp. 2333–2338.

[49] B. Mitra, F. Diaz, N. Craswell, Learning to match using local and distributed representations of text for web search, in: Proceedings of the 26th International Conference on World Wide Web, 2017: pp. 1291–1299.

[50] Y. Fan, L. Pang, J. Hou, J. Guo, Y. Lan, X. Cheng, Matchzoo: A toolkit for deep text matching, ArXiv Preprint ArXiv:1707.07270. (2017).

[51] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, ArXiv Preprint ArXiv:1412.6980. (2014).