The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Duplicate Question Identification by Integrating FrameNet with Neural Networks

Xiaodong Zhang,[1] **Xu Sun,**[1] **Houfeng Wang**[1,2]
[1] MOE Key Lab of Computational Linguistics, Peking University, Beijing, 100871, China
[2] Collaborative Innovation Center for Language Ability, Xuzhou, Jiangsu, 221009, China
{zxdcs, xusun, wanghf}@pku.edu.cn

Abstract

There are two major problems in duplicate question identification, namely lexical gap and essential constituents matching. Previous methods either design various similarity features or learn representations via neural networks, which try to solve the lexical gap but neglect the essential constituents matching. In this paper, we focus on the essential constituents matching problem and use FrameNet-style semantic parsing to tackle it. Two approaches are proposed to integrate FrameNet parsing with neural networks. An ensemble approach combines a traditional model with manually designed features and a neural network model. An embedding approach converts frame parses to embeddings, which are combined with word embeddings at the input of neural networks. Experiments on Quora question pairs dataset demonstrate that the ensemble approach is more effective and outperforms all baselines.[1]

Introduction

Duplicate question identification (DQI) aims to compare two questions and identify whether they are semantically equivalent or not, i.e., a binary classification problem. It is a vital task for community question answering (CQA). With an automatic DQI method, a CQA forum can merge duplicate questions so as to organize questions and answers more efficiently. Besides, by retrieving questions that are semantically equivalent to a question presented by a user, an automatic QA system can answer the user's question with answers of the retrieved questions.

There are two major problems in DQI, namely lexical gap (or called semantic gap) and essential constituents matching. Essential constituents of a question refer to constituents that are important to the meaning of the question. A constituent contains two parts, name and value. For example, for a question asking route, there is usually a destination constituent.

Four questions are listed below to explain the two problems. The first two questions are duplicate, and the last two are non-duplicate.

- **Q1:** What is the most populous state in the USA?
- **Q2:** Which state in the United States has the most people?
- **Q3:** How can I go downtown from the airport?
- **Q4:** How can I go downtown from the park?

First, different people tend to use different words and phrases to express the same meaning. For Q1 and Q2, although they are semantically equivalent, there are only a few overlapped words. If only considering the surface form of the questions without leveraging any semantic information, it is hard to identify that they are duplicate. This problem is called lexical gap. Second, for Q3 and Q4, although most words are overlapped and there is only one different word, they are not duplicate. There are two essential constituents, points of departure and destination, in the two questions. The departure places are different, therefore the answers for one question are useless for the other one. It is hard for a model to classify them as non-duplicate because their surface forms are so similar. We refer to this problem as essential constituents matching.

Distributed representation is an effective way to tackle the lexical gap problem. Researchers have designed various similarity features based on word embeddings (Franco-Salvador et al. 2016), or acquired representations of questions via neural networks and then calculated their similarity (Santos et al. 2015; Lei et al. 2016). Although much effort has been paid to the lexical gap problem, there is little research on the essential constituents matching problem, which is also vital to DQI. Previous approaches are generally based on similarity. Therefore they are unlikely to classify Q3 and Q4 as non-duplicate. The words and sentence patterns of the two questions are so similar that representations learned by neural networks are likely to be similar. There should be a way to model the matching of essential constituents in question pairs explicitly. The unmatched essential constituents can provide strong clues for predicting question pairs as non-duplicate.

It is non-trivial to extract essential constituents in a question. A common way is to define constituent categories by experts and label some questions by annotators to get a labeled dataset. Then, a supervised sequence labeling model can be trained on the dataset to extract essential constituents, which is similar to named entity recognition task (Sang and Meulder 2003). However, defining and labeling essential constituents in open domain are impractical, consuming too much time and funding. Fortunately, there is a correlation between essential constituents and semantic units

[1]The code is available at https://github.com/zxdcs/DQI

in a semantic parse. Hence, we parse questions using a FrameNet (Baker, Fillmore, and Lowe 1998) parser and approximate essential constituents by frames. The essential constituents matching problem is transformed into a frame matching problem. In this way, manual labeling is avoided and all we need is a frame parser, which is publicly available on the Internet.

In this paper, we use FrameNet parsing for essential constituents matching and neural networks for handling lexical gap. Two approaches are proposed to integrate FrameNet parses with neural networks, namely ensemble approach and embedding approach. In the ensemble approach, two models are trained separately and their outputs are combined. For FrameNet parses, two kinds of features on the word and the frame level are designed to measure the matching degrees of essential constituents. A gradient boosting decision tree (GBDT) (Friedman 2001) classifier is trained on these features. For neural networks, any kinds of neural networks that take two sentences as input can be used. In the embedding approach, a unified model is proposed. Each kind of frame is assigned to an embedding. The frame embeddings are concatenated with word embeddings at the input of neural networks. Consequently, the representations learned by neural networks can include essential constituents information.

Frame Parsing

The FrameNet project (Baker, Fillmore, and Lowe 1998) is a semantic database of English, which contains about 200,000 manually annotated sentences linked to more than 1,200 semantic frames. It is based on a theory of meaning called Frame Semantics (Fillmore 1976). The basic idea is that the meaning of most words can be understood on the basis of semantic frames, which are represented by three major components: frame, frame elements (FEs) and lexical units (LUs).

Table 1 lists the parse of Q3 using a FrameNet parser called SEMAFOR (Kshirsagar et al. 2015). The first column lists all words in the question. In the rest columns, each column represents a frame. The word that triggers a frame is marked by bold text, and other items represent FEs of the frame. The question contains three frames, including capability, motion and buildings. Take the *motion* frame as an example, it is evoked by LU *go* and contains three FEs, i.e., *theme*, *goal* and *source*. The FE *goal* is filled by LU *downtown*.

	Capability	Motion	Buildings
How	Entity		
can	**Capability**		
I	Event	Theme	
go		**Motion**	
downtown		Goal	
from		Source	
the			
airport			**Buildings**
?			

Table 1: FrameNet-style parsing of a question.

The resemblance of frame and essential constituent is apparent. By viewing the name and LUs of a frame as the name and value of an essential constituent, a frame can be easily converted to an essential constituent. This is the main reason why FrameNet-style parsing is used.

We find that the FrameNet parsing cannot cover all essential constituents in questions, which is because of both the incomplete coverage of FrameNet and unsatisfying performance of the parser. A major missing is some location constituents. For example, in question "What is the best travel website in Spain?", the word *Spain* is not included in any frame. To overcome this shortcoming, named entities in questions are recognized by an automatic recognizer and they are fitted into the FrameNet structure. Specifically, the word *Spain* is recognized as a geo-political entity (GPE). Hence, a frame with GPE as the name and Spain as the LU is constructed.

Neural Networks Model

Neural networks models (NNMs) with word embeddings as input are ideal models to handle the lexical gap problem. Because our focus is how to leverage frame parsing rather than proposing a novel NNM, we directly use some off-the-shelf models that perform well on similar tasks. In experiments, many different kinds of neural networks are tried. Here we only introduce a basic one as an example, that is a Siamese network (Bromley et al. 1993) consisting of two bidirectional long short-term memory (BLSTM) networks. The structure is shown in Figure 1.

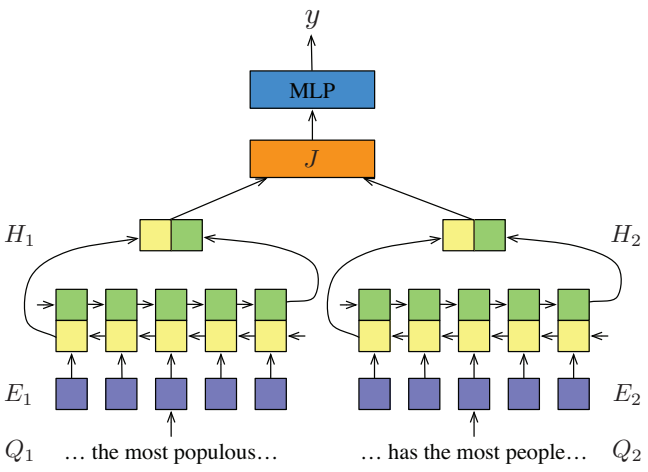

Figure 1: The structure of the BLSTM model.

At first, a question $Q = [w_1, w_2, ..., w_n]$ is mapped to an embedding matrix E via lookup table, i.e. $E = [e(w_1), e(w_2), ..., e(w_n)]$, where $e(w_t)$ is the word embedding of w_t. Then a BLSTM (Hochreiter and Schmidhuber 1997; Graves 2012) is employed to learn contextual representations of the embeddings and these representations are reduced to a fixed-length representation H. The gates, cell and output of LSTM are calculated as follows:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$
$$\hat{c_t} = tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \hat{c_t} \quad (5)$$
$$h_t = o_t \odot tanh(c_t) \quad (6)$$

where i_t, f_t, o_t are input gate, forget gate and output gate respectively, σ is a sigmoid function, W_i, W_f, W_o, W_c, U_i, U_f, U_o, U_c are weight matrices, b_i, b_f, b_c are biases, and x_t is the input at the time step t.

We use a function $LSTM$ to represent the computation of LSTM. With $e(w_t)$ as the input, the hidden state is

$$h_t = LSTM(e(w_t), h_{t-1}) \quad (7)$$

A BLSTM consists of a forward and a backward LSTM. The forward LSTM reads the input sequence as it is ordered and calculates forward hidden states $(\overrightarrow{h}_1, ..., \overrightarrow{h}_n)$. The backward LSTM reads the sequence in the reserve order and calculates backward hidden states $(\overleftarrow{h}_1, ..., \overleftarrow{h}_n)$. The fixed-length representation is obtained by concatenating the last forward and backward hidden state, i.e.,

$$H = [\overrightarrow{h}_n; \overleftarrow{h}_1] \quad (8)$$

Given two questions Q_1 and Q_2, their representations H_1 and H_2 are calculated as above-mentioned. The matching degree of H_1 and H_2 is measured by their absolute difference and element-wise product, i.e.,

$$J = [|H_1 - H_2|; (H_1 \odot H_2)] \quad (9)$$

A multi-layer perceptron (MLP) (Bishop 1995) is used to learn high-level representation and the top layer is a softmax classifier. The output is predicted scores for all classes $P^N = [p^N_+, p^N_-]$, where p^N_+ is the score of the duplicate and p^N_- is the non-duplicate.

Ensemble Approach

In this section, we describe the ensemble approach that combines the predictions of a NNM and a gradient boost decision tree (GBDT) model with features based on frame parsing. The latter model is referred to as Frame-GBDT and is described as follows.

We design two kinds of features to measure the matching degrees of essential constituents in questions, namely frame level features and lexical level features. Suppose $F = \{f_i | i \in [1, |F|]\}$ is a set of all frames in FrameNet. A question is represented as $Q = [w_1, w_2, ..., w_n]$, where n is the length of Q, and w_i is the i-th word. The frame parse of Q is $Q^P = [q_1, q_2, ..., q_n]$, where q_i is a set of frames that w_i belongs to. For the example in Table 1, w_3 is "*I*" and q_3 is {*Capability*, *Motion*}. If w_i is not assigned to any frame, the q_i is $\emptyset$. The frame level representation R is obtained by collecting all LUs in each frame. Formally,

$$R = \{r_i | i \in [1, |F|]\} \quad (10)$$
$$r_i = \{w_k | f_i \in q_k, k \in [1, n]\} \quad (11)$$

For example, if f_i is *Capability* in Table 1, r_i is {*How*, *can*, *I*}. So, R is a $|F|$-dimensional sparse vector, in which most elements are empty sets and a few elements are the LU sets of frames in a question.

For two questions Q_1 and Q_2 with frame representations R_1 and R_2, their frame matching feature is $M^R = [m^R_1, m^R_2, ..., m^R_{|F|}]$, where

$$m^R_i = \begin{cases} \frac{|r_{1i} \cap r_{2i}|}{|r_{1i} \cup r_{2i}|} + b & r_{1i} \neq \emptyset \text{ and } r_{2i} \neq \emptyset \\ 0 & r_{1i} = \emptyset \text{ and } r_{2i} = \emptyset \\ -1 & \text{otherwise} \end{cases} \quad (12)$$

That is, for a frame that exists in both questions, the matching degree is defined as Jaccard similarity[2] of the corresponding word sets plus a bias, where b is a hyper-parameter. The Jaccard similarity can be 0, thus a bias is used to distinguish with the following condition. For a frame that exists in neither question, the matching degree is 0. Otherwise, for a frame that exists in one question but not in the other one, the matching degree is -1, which indicates an unmatched frame.

An example of calculating M^R is given by the Q3 and Q4 presented in section "Introduction". We only list frames that appear at least in one question, as shown in Table 2. Suppose b is 1 and stop words are removed.

Frame	R_{Q3}	R_{Q4}	M^R
Capability	{How, can, I}	{How, can, I}	2
Motion	{I, go, downtown, from, airport}	{I, go, downtown, from, park}	1.67
Buildings	{airport}	{park}	1

Table 2: An example of calculating frame level features.

A lexical level representation is constructed in a similar way. Suppose $V = \{v_i | i \in [1, |V|]\}$ is the vocabulary. The lexical level representation is defined as

$$L = \{l_i | i \in [1, |V|]\} \quad (13)$$
$$l_i = \left\{ \bigcup_{w_j = v_i} q_j \middle| j \in [1, n] \right\} \quad (14)$$

Take Table 1 as an example again. If v_i is "*I*", l_i is {*Capability*, *Motion*}. A word can occur more than once in a question. Here we use a simple way, that is, merging the frame sets of all its occurrences. Hence, L is a $|V|$-dimensional sparse vector like R.

The lexical matching feature $M^L = [m^L_1, m^L_2, ..., m^L_{|V|}]$ is calculated in a similar way as M^R. The only difference is that the r_{1i} and r_{2i} in Equation (12) are changed to l_{1i} and l_{2i}. An example for calculating M^L of Q3 and Q4 is demonstrated in Table 3. Suppose b is also 1.

A final matching feature vector M is constructed by concatenating the frame and lexical matching features, i.e. $M = [M^R; M^L]$. A GBDT classifier is adopted to train the essential constituents matching model with M as features. We

[2] https://en.wikipedia.org/wiki/Jaccard_index

LU	L_{Q3}	L_{Q4}	M^L
How	{capability}	{capability}	2
can	{capability}	{capability}	2
I	{capability, motion}	{capability, motion}	2
go	{motion}	{motion}	2
downtown	{motion}	{motion}	2
from	{motion}	{motion}	2
airport	{motion, buildings}	$\emptyset$	-1
park	$\emptyset$	{motion, buildings}	-1

Table 3: An example of calculating lexical level features.

also try other classifiers, like logistic regression and support vector machines, but find that GBDT performs best. The output is predicted scores for all classes $P^G = [p^G_+, p^G_-]$.

We use a straightforward but effective ensemble method. The Frame-GBDT and NNM are combined by a weighted average, and the predicted label is the class with max score in the combined score. Formally, we have

$$P = \alpha P^N + (1 - \alpha) P^G \tag{15}$$

$$y = \arg\max_{i \in \{+,-\}} P_i \tag{16}$$

where $\alpha \in [0, 1]$ is a hyper-parameter. With α becomes larger, the combined result is more prone to the prediction of the NNM, otherwise it is more prone to the Frame-GBDT.

Embedding Approach

In the ensemble approach, two models are used to solve the two challenges in DQI separately, and their outputs are combined to make a final prediction. In this section, we present a unified NNM to solve those two challenges together. To leverage frame information, each frame is mapped to an embedding. The frame embeddings can be combined with word embeddings as inputs, providing more information to learn better representation. Figure 2 demonstrates the process of representation learning. The processes of calculating matching degree of two representations and making prediction are omitted here.

Given a question $Q = [w_1, w_2, ..., w_n]$, the word embedding matrix $E^w = [e(w_1), e(w_2), ..., e(w_n)]$ is computed as before. We use a superscript to distinguish word and frame embeddings. The frame parse of Q is $Q^P = [q_1, q_2, ..., q_n]$ as described in the section "Ensemble Approach". The only difference is that the words that are not assigned to any frame are allocated with a manually added frame called "N/A", which also corresponds to an embedding. This is meant to make every word in a question has at least a frame embedding for further computation. A word can be assigned to more than one frame. The frame representation $c(w_t)$ of a word w_t is calculated by averaging all its frame embeddings. Formally, we have

$$c(w_t) = \frac{1}{|q_t|} \sum_{f \in q_t} g(f) \tag{17}$$

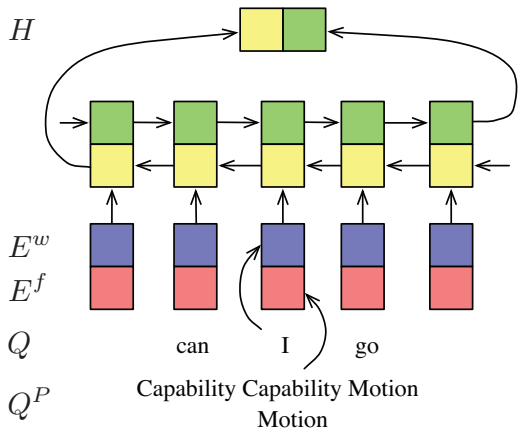

Figure 2: The structure of the embedding approach.

where $g(f)$ is the embedding of the frame f. Thus, the frame representation of a question $E^f = [c(w_1), c(w_2), ..., c(w_n)]$ is obtained. The input x_t of BLSTM at the time step t is the combination of the word embedding and the frame representation of w_t, i.e.,

$$x_t = [e(w_t); c(w_t)] \tag{18}$$

Therefore, the hidden state of LSTM is calculated as

$$h_t = LSTM([e(w_t); c(w_t)], h_{t-1}) \tag{19}$$

In this way, the frame information is incorporated into a NNM. The learned representation can include frame information so that the comparison of two representations can imply the comparison of essential constituents in two questions.

Experiments

Dataset

The recently released Quora question pairs (QQP) dataset[3] is adopted in our experiments. It consists of over 400,000 potential question duplicate pairs. Each sample contains two questions and a label. Like the four questions in the section "Introduction", Q1 and Q2 is a pair and the label is positive, while Q3 and Q4 is a pair and the label is negative. Because there is not an official partition of train/dev/test set, we shuffle the dataset randomly and split train/dev/test set with a proportion of 8:1:1. The statistics of the question pairs are listed in Table 4.

	Positive	Negative	Total
Train (80%)	119282	204150	323432
Dev (10%)	15010	25419	40429
Test (10%)	14971	25458	40429
Total	149263	255027	404290

Table 4: Statistics of QQP dataset.

[3]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

Experimental Setup

We choose some well-performed methods on other similar tasks for comparison and reimplement them on the QQP dataset.

- **BM25** (Robertson and Zaragoza 2009) is a widely used similarity for information retrieval, especially in search engines.
- **SimBow** (Franco-Salvador et al. 2016) is the top team of question-to-question similarity ranking task in SemEval 2016. They use lexical and semantic similarity measures.
- **BLSTM** is the model presented in the section "Neural Networks Model".
- **CNN** has the same overall structure with the BLSTM method except that the representations of questions are learned by CNN (LeCun et al. 1998) with max-pooling.
- **CNN-II** is the ARC-II model proposed by Hu et al. (2014). It directly compares each pair of CNN representations, instead of compressing them to fixed-length first.
- **AP** (Santos et al. 2016) uses a two-way attention mechanism to calculate the matching relation of representations of two sentences.
- **DA** (Parikh et al. 2016) decomposes natural language inference problem into three steps, i.e. attend, compare and aggregate.
- **AI** (Zhang et al. 2017) improves AP by modeling the interaction of representations and using more information to calculate the attention.
- **Frame-GBDT** is our proposed model in the section "Ensemble Approach". It can deal with the essential constituents matching problem.

For AP, DA and AI, there are some variants based on which kind of neural network is utilized to learn representations. CNN and BLSTM are commonly used. Consequently, there are actually six methods, namely **AP-CNN**, **AP-BLSTM**, **DA-CNN**, **DA-BLSTM**, **AI-CNN** and **AI-BLSTM**.

All hyper-parameters are tuned on the development set. We use spaCy[4] to recognize named entities, and the full list of entity types can be found at the documents[5] of spaCy. LightGBM[6] is used for GBDT implementation. The maximum number of leaves in a tree is 700 and minimal number of data in a leaf is 0. The number of boosting round is 10000 and the early stopping round is 100. All neural networks are implemented using PyTorch[7]. Word embeddings[8] have the dimension of 300 and are pretrained by GloVe (Pennington, Socher, and Manning 2014). Frame embeddings are 50-dimensional, which are initialized randomly with a uniform distribution between $[-1, 1]$. Word and frame embeddings are both fine-tuned at the training process. The dimension of H is 300, thus the dimension of J is 600. The MLP consists of a 200-dimensional hidden layer. The model is trained using Adam (Kingma and Ba 2014) optimization method with the learning rate set to 0.001. The batch size is set to 100.

[4]https://spacy.io/

[5]https://spacy.io/docs/usage/entity-recognition

[6]https://github.com/Microsoft/LightGBM

[7]http://pytorch.org/

[8]http://nlp.stanford.edu/data/glove.840B.300d.zip

Experimental Results

We set up two groups of experiments. The first group is about the ensemble approach, that is to see whether the performance of baselines can be improved by combining with the Frame-GBDT. The results are listed in Table 5.

Method	Baseline accuracy	Improved accuracy (+ Frame-GBDT)
BM25	65.23	86.12 (+20.89)
SimBow	67.96	86.30 (+18.34)
BLSTM	86.17	87.92 (+1.75)
CNN	85.93	87.69 (+1.76)
CNN-II	84.69	87.27 (+2.58)
AP-CNN	85.28	87.40 (+2.12)
AP-BLSTM	86.11	87.84 (+1.73)
DA-CNN	84.50	87.14 (+2.64)
DA-BLSTM	85.72	87.62 (+1.90)
AI-CNN	86.55	87.97 (+1.42)
AI-BLSTM	87.43	**88.53** (+1.10)

Table 5: Results of the ensemble method. The second column is the accuracy of baseline methods. The third column is the accuracy of our proposed ensemble model, i.e., baseline + Frame-GBDT. The improvements are listed in the parentheses.

The BM25 and SimBow perform badly and are discussed in detail in the next paragraph. For the NNMs, we can see that a sophisticated model is not necessarily better than a straightforward one. The results of CNN-II, AP and DA are comparable or worse than BLSTM and CNN. Nevertheless, AI-BLSTM outperforms other methods by a large margin, demonstrating strong capability on this task. Next, we combine these methods with Frame-GBDT, as shown in the third column. It is worth mentioning that the Frame-GBDT achieves an accuracy of 86.01% individually, which is comparable to most NNMs and it demonstrates the effectiveness of the designed features. With Frame-GBDT added, all neural network models are improved by $1 \sim 3\%$[9]. The improvements are due to the fact that Frame-GBDT is good at essential constituents matching and neural network models are good at handling lexical gap. The two kinds of models are complementary so that the ensemble model can deal with the two problems together. By combining AI-BLSTM and Frame-GBDT, we can achieve an accuracy of 88.53% on this task.

BM25 and SimBow perform much worse than other methods. However, these two methods perform well in question-to-question ranking (QR) task in SemEval 2016 (Nakov et al. 2016). The QR task is similar to the DQI task, both judging the relation of two questions, so it is worth exploring the reason of the difference. Through statistical analysis, we find that the main difference of question pairs in the two datasets is word overlap rate (WOR). WOR of two questions is defined as the number of overlapped words between them divided by the average length of them. Hence, WOR is

[9]All improvements are statistically significant ($P < 0.05$).

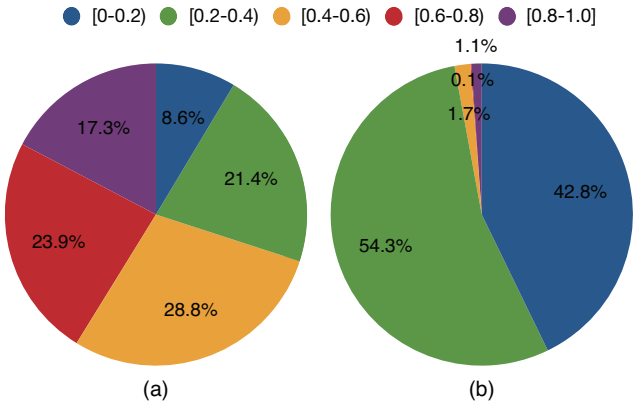

Figure 3: WOR of QQP (left) and QR (right) dataset.

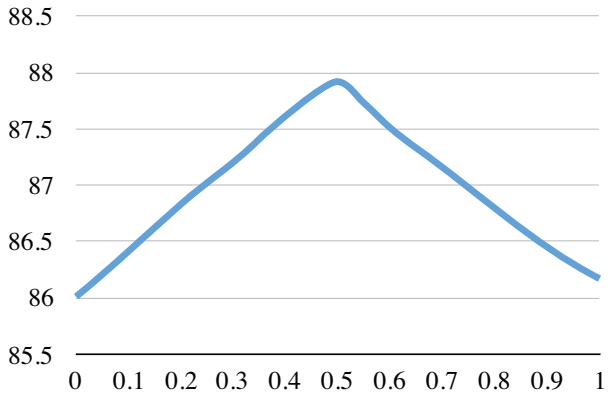

Figure 4: The influence of weight in ensemble model.

a real number between 0 and 1. We calculate WOR of question pairs in the two datasets and the proportion is shown in Figure 3. High WOR pairs take a large proportion in QQP dataset, which means the difficult point is identifying non-duplicate in the alike pairs. The word similarity based methods BM25 and SimBow are likely to fail in such case. In QR dataset, most pairs have a low WOR, indicating that the key point is to discover connection between the unlike pairs and these two methods work well.

The second group of experiments is about the embedding approach. We only compare four BLSTM based methods, because the first group has demonstrated that the results of BLSTM based methods are comparable or better than CNN based methods. The results are listed in Table 6.

Method	Baseline accuracy	Improved accuracy (+ Frame embedding)
BLSTM	86.17	86.42 (+0.25)
AP-BLSTM	86.11	86.22 (+0.11)
DA-BLSTM	85.72	86.03 (+0.31)
AI-BLSTM	87.43	87.62 (+0.19)

Table 6: Results of the embedding method. The second column is the accuracy of baseline methods. The third column is the accuracy of these models with frame embeddings added. The improvements are listed in the parentheses.

With the frame embedding added, the four methods have some improvements. However, the improvements are marginal compared to the ensemble approach. We think it is because the continuous representation is not suitable for essential constituents matching. Take the following two questions as an example: "Which is the largest city in America?" and "Which is the largest city in Canada?". With frame embeddings, the neural network can be aware that America and Canada are location constituents. However, in this approach, the essential constituents matching is performed in continuous space. The word embeddings of America and Canada are normally similar. Consequently, the model cannot discover that the locations are unmatched. In the Frame-GBDT, the essential constituents matching is performed in discrete space. America and Canada are unmatched because they are different words. The two groups of experiments demonstrate that the discrete matching is more suitable for essential constituents matching than the continuous way.

Further Analysis of the Ensemble Approach

In this subsection, we provide some further analysis of the ensemble approach, mainly about the combination of Frame-GBDT and NNM.

Two models are combined by a weighted average as in Equation (15) and α is the weight. Figure 4 shows the change of performance with the change of α. Here we choose BLSTM as the NNM. The combined model becomes Frame-GBDT when α is 0, and becomes BLSTM when α is 1. When α is close to 0.5, the model performs best, and the score decreases nearly linearly as α becomes larger or smaller. It indicates that the two separate models contribute equally to the combined model. Therefore, the information learned by the two models is different, which supports our argument that essential constituents matching and lexical gap are both important in DQI and better performance can be achieved by combining them.

As is known to all, ensemble methods can improve the performance. Hence, is the improvement of the feature-based method just because of combining two models? The answer is negative. We combine different kinds of models to illustrate it, as shown in Table 7.

Method	Accuracy (%)
BLSTM	86.17%
BLSTM + CNN	87.14%
BLSTM + AP-BLSTM	87.01%
BLSTM + Frame-GBDT	87.92%

Table 7: Results of combining two models.

The BLSTM is selected as the baseline model and three other models are used for combination, including CNN, AP-BLSTM and Frame-GBDT. These three models are selected

because their scores are close, which ensures a fair comparison. When BLSTM is combined with CNN or AP-BLSTM, the improvement is about 1%. However, when it is combined with Frame-GBDT, the improvement is close to 2%. The difference indicates that the information learned by NNMs is basically same so that the improvement is limited. Even if the score of Frame-GBDT is a little lower than CNN and AP-BLSTM, it can learn different information and the improvement of combination is more significant.

Related work

With the rapid development of CQA forum, DQI has become a hot research area in recent years. DQI has tight connection with text relevance and paraphrase detection. Early work on text relevance calculation usually uses similarity measures based on word overlap (Broder 1997; Wu, Zhang, and Huang 2011). However, these methods are not capable of measuring semantic equivalence. More sophisticated methods use machine translation, knowledge graphs, and topic model for better measuring text similarity (Jeon, Croft, and Lee 2005; Zhou et al. 2013; Ji et al. 2012). Ji and Eisenstein (2013) reweight term matrix and use the latent representation from matrix factorization for paraphrase classification. In recent years, many deep learning models have been proposed to avoid feature engineering and handle lexical gap. Hu et al. (2014) propose two CNN architectures for matching sentences. ARC-I first finds the representation of each sentence and then compares their representations, while ARC-II calculates the interaction of two sentences before obtaining their representations. Some follow-up work adopts the idea of ARC-II and sophisticated methods are proposed to calculate the interaction of two sentences, mainly based on attention mechanism (Santos et al. 2016; Parikh et al. 2016; Zhang et al. 2017).

There has been some prior work on question similarity calculation. Santos et al. (2015) combine a CNN and a bag-of-words representation for comparing questions. Barzilay et al. (2016) propose a semi-supervised method that pretrains a gated convolution model within an encoder-decoder framework. Wang, Hamza, and Florian (2017) propose a bilateral multi-perspective matching model and achieve good performance on DQI. There have been evaluations on question and answer retrieval in SemEval for several years (Nakov et al. 2016). For the question-to-question ranking subtask, Franco-Salvador et al. (2016) use both lexical and semantic-based similarity measures and take the first place. These methods have made great progress in solving lexical gap, but they have not considered the essential constituents matching problem.

Essential constituents is close to the notion of *essential question terms* proposed by Khashabi et al. (2017). The main difference is that essential question terms only define which words are essential, while essential constituents further define the name (category) of constituents so that matching under each category can be performed. Extracting essential constituents in questions is similar to spoken language understanding (SLU) task, which identifies intents and slots (can be view as essential constituents) in utterances (Zhang and Wang 2016). However, SLU usually needs to define and label slots, which is impractical in open-domain CQA. In this paper, the essential constituents matching problem is transformed into frame matching to avoid defining and labeling essential constituents.

Conclusion

In this paper, we point out the importance of essential constituents matching in DQI. Frames are adopted to approximate essential constituents so as to avoid labeling essential constituents. Two approaches are proposed to leverage the frame information. The ensemble approach combines a GBDT model with designed frame features and a neural networks model. The embedding approach represents frames as embeddings, which are combined with word embeddings at the input of neural networks. Experiments demonstrate that the ensemble method is more effective. The Frame-GBDT model can handle essential constituents matching and the neural network model can deal with lexical gap. In future work, we plan to explore other kinds of semantic parsing, such as abstract meaning representation and NomBank.

Acknowledgments

Our work is supported by the National Key Research and Development Program of China under Grant No.2017YFB1002101 and National Natural Science Foundation of China under Grant No.61572049. We thank the anonymous reviewers for helpful comments. The corresponding authors of this paper are Houfeng Wang and Xu Sun.

References

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING)*, 86–90.

Barzilay, T. L. H. J. R.; Jaakkola, T.; Tymoshenko, K.; and Marquez, A. M. L. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 1279–1289.

Bishop, C. M. 1995. *Neural networks for pattern recognition*. Oxford University Press.

Broder, A. Z. 1997. On the resemblance and containment of documents. In *Proceedings of the Compression and Complexity of Sequences*, 21–29.

Bromley, J.; Bentz, J. W.; Bottou, L.; Guyon, I.; LeCun, Y.; Moore, C.; Säckinger, E.; and Shah, R. 1993. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence* 7(4):669–688.

Fillmore, C. J. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences* 280(1):20–32.

Franco-Salvador, M.; Kar, S.; Solorio, T.; and Rosso, P. 2016. Uh-prhlt at semeval-2016 task 3: Combining lexi-

cal and semantic-based features for community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 814–821.

Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* 1189–1232.

Graves, A. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hu, B.; Lu, Z.; Li, H.; and Chen, Q. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems (NIPS)*, 2042–2050.

Jeon, J.; Croft, W. B.; and Lee, J. H. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM)*, 84–90.

Ji, Y., and Eisenstein, J. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, 891–896.

Ji, Z.; Xu, F.; Wang, B.; and He, B. 2012. Question-answer topic model for question retrieval in community question answering. In *Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM)*, 2471–2474.

Khashabi, D.; Khot, T.; Sabharwal, A.; and Roth, D. 2017. Learning what is essential in questions. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL)*, 80–89.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kshirsagar, M.; Thomson, S.; Schneider, N.; Carbonell, J. G.; Smith, N. A.; and Dyer, C. 2015. Frame-semantic role labeling with heterogeneous annotations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and The 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, 218–224.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Lei, T.; Joshi, H.; Barzilay, R.; Jaakkola, T. S.; Tymoshenko, K.; Moschitti, A.; and i Villodre, L. M. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 1279–1289.

Nakov, P.; i Villodre, L. M.; Moschitti, A.; Magdy, W.; Mubarak, H.; Freihat, A. A.; Glass, J.; and Randeree, B. 2016. Semeval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 525–545.

Parikh, A. P.; Täckström, O.; Das, D.; and Uszkoreit, J. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2249–2255.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, volume 14, 1532–1543.

Robertson, S., and Zaragoza, H. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval* 3(4):333–389.

Sang, E. F. T. K., and Meulder, F. D. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at NAACL-HLT*, 142–147.

Santos, C. d.; Barbosa, L.; Bogdanova, D.; and Zadrozny, B. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, volume 2, 694–699.

Santos, C. d.; Tan, M.; Xiang, B.; and Zhou, B. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Wang, Z.; Hamza, W.; and Florian, R. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, 4144–4150.

Wu, Y.; Zhang, Q.; and Huang, X. 2011. Efficient near-duplicate detection for q&a forum. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, 1001–1009.

Zhang, X., and Wang, H. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, 2993–2999.

Zhang, X.; Li, S.; Sha, L.; and Wang, H. 2017. Attentive interactive neural networks for answer selection in community question answering. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 3525–3531.

Zhou, G.; Liu, Y.; Liu, F.; Zeng, D.; and Zhao, J. 2013. Improving question retrieval in community question answering using world knowledge. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 13, 2239–2245.