Match²: A Matching over Matching Model for Similar Question Identification

Zizhen Wang*,†, Yixing Fan†, Jiafeng Guo*,†, Liu Yang‡, Ruqing Zhang‡, Yanyan Lan*†‡, Xueqi Cheng*†‡, Hui Jiang‡§, Xiaozhao Wang‡§

{wangzizhen,fanyixing,guojiafeng,zhangruqing,lanyanyan,cxq}@ict.ac.cn
lyang@cs.umass.edu, {huijiang,jh,orlando}@alibaba-inc.com

*University of Chinese Academy of Sciences, Beijing, China
†CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
‡University of Massachusetts Amherst, Massachusetts, United States
§Alibaba Group, Beijing, China

ABSTRACT
Community Question Answering (CQA) has become a primary means for people to acquire knowledge, where people are free to ask questions or submit answers. To enhance the efficiency of the service, similar question identification becomes a core task in CQA which aims to find a similar question from the archived repository whenever a new question is asked. However, it has long been a challenge to properly measure the similarity between two questions due to the inherent variation of natural language, i.e., there could be different ways to ask a same question or different questions sharing similar expressions. To alleviate this problem, it is natural to involve the existing answers for the enrichment of the archived questions. Traditional methods typically take a one-side usage, which leverages the answer as some expanded representation of the corresponding question. Unfortunately, this may introduce unexpected noises into the similarity computation since answers are often long and diverse, leading to inferior performance. In this work, we propose a two-side usage, which leverages the answer as a bridge of the two questions. The key idea is based on our observation that similar questions could be addressed by similar parts of the answer while different questions may not. In other words, we can compare the matching patterns of the two questions over the same answer to measure their similarity. In this way, we propose a novel matching over matching model, namely Match², which compares the matching patterns between two question-answer pairs for similar question identification. Empirical experiments on two benchmark datasets demonstrate that our model can significantly outperform previous state-of-the-art methods on the similar question identification task.

CCS CONCEPTS
• Information systems → Question answering

KEYWORDS
community question answering, similar question identification; matching over matching

ACM Reference Format:

1 INTRODUCTION
Community Question Answering (CQA) services, such as StackExchange¹ and Quora², have grown in popularity in recent years as a platform for people to exchange knowledge. In CQA, users can ask their questions or submit answers to questions in a collaborative fashion. Although CQA services greatly benefit users with high-quality human-generated answers for solving their problems, the efficiency becomes a big concern as the asker need to wait until someone submits the answer to his/her question. To alleviate this problem, similar question identification becomes a core task in CQA which aims to find a similar question from the archived repository whenever a new question is proposed. In the meantime, similar question identification could also help reduce redundant questions in CQA services, saving a lot of users’ efforts.

However, it has long been a challenge to properly measure the similarity between two questions, which are usually very short in length, due to the inherent variation of natural language. On one hand, there could be different ways to express the same question, leading to the lexical gap [26, 33]. For example, as shown in Figure 2 Case A, the user question \( Q^u \) and the archived question \( Q^a \) are similar and could be addressed by the archived answer \( A^u \) of \( Q^u \), but they have very different expressions. On the other hand, there could be different questions sharing very similar expressions, leading to false positive predictions if one cannot distinguish their subtle difference. For example, as shown in Figure 2 Case B, although these two questions share many words in common, their focus is totally

¹https://stackexchange.com/
²https://www.quora.com/
Q1: What could it mean to rotate a distribution?
Q2: How to understand moments for a random variable?
A1: The name of moment comes from physics. Consider the distribution of a random variable as linear rod, the center of gravity is the first moment, and the moment of rotational inertia about the center of gravity is the variance.

Case A:
Q1: What is the method for the Marine shatter step?
Q2: What is the optimal Marine shatter step timing?
A1: The base attack of Marines is 0.8608 seconds, and 0.57387 seconds while stimulated.
The game speed factor is about 1.4. The random delay between attacks ranges from 0.135 to 0.605 seconds. Based on the above numbers, we get the optimal timing in worst unstimmed case is 0.7841 seconds, and 0.4992 seconds while stimulated.
Label: 1

Case B:
Q1: What is the correct method for the Marine shatter step?
Q2: What is the optimal Marine shatter step timing?
A1: Actually, due to the memory management strategies in Java, object creation can be considered free compared to everything else in JVM. The other part of the cost is object destruction. The modern garbage collector algorithm deallocates when free memory is needed in a certain generation. If the JVM heap is big enough, then no deallocations happen long enough to cause any pauses.
Label: 0

Case C:
Q1: What is the garbage collector in Java?
Q2: Should we avoid object creation in Java?
A1: Considered free compared to everything else in JVM. The other part of the cost is object destruction. The modern garbage collector algorithm deallocates when free memory is needed in a certain generation. If the JVM heap is big enough, then no deallocations happen long enough to cause any pauses.
Label: 0

Figure 2: The cases from StackExchange. The blue and yellow parts denote the focuses of the question and the corresponding related answer parts. The red parts denote the answer text which can address both the user question and the archived question. The archived answer is helpful to identify the question similarity (Case A,B), but it may introduce unexpected noises (Case C).

different (one about the “method” and one about the “timing”), and thus could not be addressed by the same answer.

Similar question identification has attracted extensive studies in recent years. Some early works in this direction formulated it as a question-question matching problem, as shown in Figure 1(a). Both conventional machine learning methods [7, 18, 46, 47] and deep neural networks [9, 11, 15, 31, 33, 41, 48] have been applied to this problem. However, simply based on two questions, even most advanced neural models cannot well address the two challenges mentioned above due to the sparse information in questions.

Since archived questions usually associated with answers, it is natural to involve the existing answers for the enrichment of the archived questions to alleviate the sparsity problem. To leverage the answers of the archived question, existing methods typically take a one-side usage, as shown in Figure 1(b), which treats the answer as some expanded representation of the corresponding question. For example, Ji et al. [26] employed the archived answer to learn an enriched topic representation of questions for similarity computation. Gupta et al. [16] matched the user question to the archived question and its answer separately then aggregated them with an attention mechanism. Unfortunately, the one-side usage may introduce unexpected noises into the similarity computation, leading to inferior performance. The reason is that answers are not equivalent representations of the corresponding questions. Answers are often long and cover diverse topics/aspects that may be beyond the scope of the corresponding question. For example, as shown in Figure 2 Case C, these two questions, one about garbage collector and one about object creation, are different in semantics. However, if we simply expand the archived question $Q^a$ with its answer $A^a$ which also talks about the garbage collector, we are prone to predict that these two questions are similar which is apparently a false positive prediction.

In fact, if we look at these cases carefully, we may find the following observation: similar questions could be addressed by similar parts of the answer while different questions may not. For example, as shown in Figure 2 Case A, these two questions are similar since they both can be addressed/connected by the similar parts of the archived answer $A^a$. However, in Case C, although the archived answer may be related to both questions, the related parts are quite different for the two questions. These cases show that it is not how similar the archived answer to the user question decides the question similarity. It is how the archived answer matches the two questions contributes to the similarity of the two questions. Therefore, we argue that the archived answer should not be simply viewed as a bridge of the two questions, namely a two-side usage in this work as depicted in Figure 1(c).

Based on the above idea, we propose a novel Matching over Matching Model, namely Match$^2$ for short, which compares the matching patterns of the two questions over the same answer for similar question identification. Specifically, Match$^2$ contains three modules, including the Representation-based Similarity Module, the Matching Pattern-based Similarity Module and the Aggregation module. The Representation-based Similarity module is similar to previous question matching methods, which generates a similarity vector between two questions simply based on their representations. The major enhancement is the Matching Pattern-based Similarity module. This module has a Siamese Network structure, which takes two question-answer pairs as the inputs, learns their matching patterns separately, builds a matching similarity tensor by comparing the
two matching patterns, and finally produces the similarity vector between the questions by compressing the matching similarity tensor. Both the representation-based and matching pattern-based similarity vectors are aggregated in the Aggregation module to produce the final identification prediction. The Aggregation module adopts a gate mechanism which takes the representation-based similarity as the primary one and the matching pattern-based similarity as the complementary one for the final decision. A multi-task learning strategy is employed to train the Match\textsuperscript{2} model.

We evaluate the effectiveness of the proposed Match\textsuperscript{2} model based on two widely used CQA benchmarks, i.e., CQADupStack [21] and QuoraQP\textsuperscript{3}. To incorporate the answer information in QuoraQP, we crawled archived answers of the corresponding questions from Quora and enrich the benchmark into a new answer-expanded version, namely QuoraQP\textsuperscript{-a}. The experimental results on these two benchmarks demonstrated that our method can significantly outperform those state-of-the-art methods on the similar question identification task.

The major contributions of this paper include:

1. We analyze the role of the archived answer in the similar question identification task and propose a two-side usage of the answer which leverages it as a bridge of the two questions.
2. We propose a novel matching over matching (Match\textsuperscript{2}) model to compare the matching patterns of the two questions over the same answer for similar question identification.
3. We conduct extensive comparisons and analysis against the state-of-the-art similar question identification models on benchmarks to demonstrate the effectiveness of our proposed method.

## 2 RELATED WORK

In this section, we briefly review the most related topics to our work in CQA, i.e., question matching. Question matching which evaluates the similarity between two questions, could be further divided into the question deduplication task and the similar question identification task with regard to different application scenarios.

### 2.1 Question Deduplication

Question deduplication aims to merge or remove the redundant questions in the archived question threads. Early studies mainly focused on designing effective features to measure the similarities between two questions, such as lexical features [4, 17, 23], syntactic features [8, 30, 42], or heuristic features [3, 13]. Many recent successes on this task have been achieved by advanced neural network models. For example, Pang et al. [32] evaluated the question similarity from hierarchical levels. Wan et al. [41] modeled the recursive structure between question pairs with spatial RNN. Tay et al. [38] proposed a CSRAN model to learn fine-grained question matching details. Yang et al. [48] built RE2 model with stacked alignment layers to keep the model fast while still yielding strong performance, and Devlin et al. [11] pre-trained a stacked transformer network which can be used for question deduplication task after fine-tuning.

Besides, the question threads in the community include not only the question texts but also other information, e.g., topics, comments and answers, which provide other perspectives for question deduplication. Zhang et al. [51] proposed a topic model approach to take answer quality into account. Wu et al. [45] proposed the QCN network to make use of the subject-body relationship of the community questions. Filice et al. [13] proposed a method to utilize the interconnection information between the question and its comments. Liang et al. [27] employed adaptive multi-attention mechanism to enhance questions with their corresponding answers. Moreover, many researchers have considered the use of different kinds of external resources. Wu et al. [44] employed various types of handcraft features to measure the question semantic similarity. Zhou et al. [54] used the semantic relations extracted from the global knowledge of Wikipedia\textsuperscript{4}.

### 2.2 Similar Question Identification

Similar question identification aims to find a similar question from the archived repository for a new question issued by a user. It is usual to frame the similar question identification as a retrieval task where the user question is taken as a query and archived questions are ranked based on their semantic similarities to the query. Hence, classical retrieval methods, e.g., BM25 [34] and LMR [50], have been applied for this task. There are also researchers employed statistical translation [25, 46, 52], topic model [5, 55] and relation extraction methods [35] to identify the similar questions. Recently, deep learning methods have been widely adopted to solve it. For example, Qiu et al. [33] employed convolutional neural network to encode questions in semantic space. Wan et al. [40] proposed MV-LSTM to capture the contextualized local information with multiple positional question representations. Furthermore, many works considered the use of different kinds of complementary information, such as question category [6, 12, 53], Wikipedia concepts [2] and corresponding answer [16, 26, 36].

Even some of the researchers on similar question identification have focused on ranking models, they might face the computational complexity and evaluation difficulty problem [20]. To address this issue, many works model the task as a classification task, which aims to explicitly predict whether the archived question is similar with the user question or not. For example, Wang et al. [43] employed a bilateral mechanism to enhance single direction matching. Chen et al. [9] proposed a sequential inference model based on chain LSTMs for the recursive matching architectures. Gong et al. [15] used DenseNet [22] to hierarchically extract semantic features from questions interaction space. Hoogeveen et al. [20] adopted meta data such as user features to identify the question relation. It seems that some models could not only be applied to similar question identification but also question deduplication task, but we can find the clear difference, i.e., the user question in similar question identification has few information except the text itself.

## 3 OUR APPROACH

In this section, we present the Matching over Matching (Match\textsuperscript{2}) model for the similar question identification task in detail. We first

\[1]\text{https://www.kaggle.com/c/quora-question-pairs}\]

\[2]\text{https://en.wikipedia.org/wiki/Main_Page}\]
Table 1: A summary of key notations in this work.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p_u, p_a)</td>
<td>The matching pattern of the user question and the archived question over the archived answer</td>
</tr>
<tr>
<td>(P)</td>
<td>The pattern similarity tensor</td>
</tr>
<tr>
<td>(S, S, s)</td>
<td>The pattern similarity function at tensor-, layer- and element-wise</td>
</tr>
<tr>
<td>(v_q)</td>
<td>The representation-based similarity vector</td>
</tr>
<tr>
<td>(v_a)</td>
<td>The matching pattern-based similarity vector</td>
</tr>
<tr>
<td>(v)</td>
<td>The question similarity vector</td>
</tr>
<tr>
<td>(r)</td>
<td>The main task loss ratio</td>
</tr>
</tbody>
</table>

We adopt \(B\) which starts with a transformer layer. This stacked structure has two types of outputs, (1) matching pattern layer: to compare the matching patterns of the two questions over the same answer; (2) pattern similarity layer: to build a pattern similarity tensor by contrasting the two matching patterns; and (3) compression layer: to produce the similarity vector between the questions by compressing the matching similarity tensor.

3.3.1 Matching Pattern Layer. We adopt Bert again to compute the matching patterns of the two questions over the same answer. Different from the \(B^p\) from Equ.1, we use the sequence features \(B^s\) here and divide it into two parts to represent the question and answer respectively.

Take the matching pattern \(P^u\) between the user question \(Q^u\) and the archived answer \(A^a\) as an example. Firstly, the user question \(Q^u\) with a sequence of \(m\) tokens is represented by concatenating the question sequence features from each transformer layer,

\[
\hat{Q}^u = [\hat{Q}^u_1, \ldots, \hat{Q}^u_l, \ldots, \hat{Q}^u_L],
\]

where \(L\) is the number of transformer layers in Bert, \(\hat{Q}^u_l \in \mathbb{R}^{H \times m}\) is the \(l\)-th question sequence feature separated from \(B^s\). In the same way, the archived answer \(A^a\) that has \(w\) tokens is represented as \(\hat{A}^a_l \in \mathbb{R}^{L \times H \times w}\). Finally, the layer-wise matching pattern \(P^u_l\) between \(Q^u\) and \(A^a\) in the \(l\)-th transformer layer is computed as,

\[
P^u_l = \hat{Q}^u_l \odot \hat{A}^a_l.
\]

Hence, by concatenating the \(L\) layer-wise matching patterns, we can obtain the final matching pattern \(P^u\) between the user question \(Q^u\) and the archived answer \(A^a\), i.e.,

\[
P^u = [P^u_1, P^u_2, \ldots, P^u_L] \in \mathbb{R}^{L \times m \times w}.
\]

The matching pattern \(P^u\) of the archived question answer pair can be computed in the same way as described above. It should be noted that the Bert architecture in this module does not share the parameters with that used in Section 3.2.

3.3.2 Pattern Similarity Layer. In this layer, we compute a pattern matching similarity tensor \(P^s\) given the two matching patterns \(P^u\) and \(P^a\), i.e.,

\[
P^s = S(P^u, P^a) \in \mathbb{R}^{L \times m \times n},
\]

where \(S\) denotes the tensor-wise similarity function, and \(P^s_i\) denotes the layer-wise matching pattern which is defined as,

\[
P^s_i = S(P^u_i, P^a_i) \in \mathbb{R}^{m \times n}.
\]

Specifically, the element-wise matching pattern similarity scalar \(p^s_{iij}\) is computed by,

\[
p^s_{iij} = s(p^u_{iij}, p^a_{iij}),
\]

where \(p^u_{iij}\) is the matching pattern from the \(j\)-th token in the user question to the archived answer, as well as \(p^a_{iij}\) represents that from the \(j\)-th archived question token.

Here, we propose five element-wise similarity functions \(s(x, y)\) to compute the similarity between a question and an answer.

- **Dot product** between two vectors is based on the projection of one vector onto another, which is defined as follows:

\[
s_{dot}(x, y) = <x, y> = x^T y.
\]
\[ s_{\text{cos}}(x, y) = \frac{\langle x, y \rangle}{||x|| \cdot ||y||} \]

where \( || \cdot || \) denote the L2 norm of vector.

- **L1** [24] represents the similarity based on Manhattan distance between vectors as follows,

\[ s_{l1}(x, y) = \frac{1}{1 + \sum_{t=1}^{T} |x_t - y_t|} \]

- **L2** is another widely used distance-based similarity function. Different from the L1 function, it is based on euclidean distance, namely,

\[ s_{l2}(x, y) = \frac{1}{1 + \sqrt{\sum_{t=1}^{T} (x_t - y_t)^2}}. \]

- **Jesene-Shannon** [24] firstly transforms the vector to a distribution with \textit{softmax} function, and then quantifies their difference by Jesene-Shannon Divergence [14].

\[ s_{\text{JSD}}(x, y) = 1 - JSD(\text{softmax}(x), \text{softmax}(y)). \]

### 3.3.3 Compression Layer

The compression layer aims to produce the matching pattern-based similarity vector by compressing the pattern similarity tensor \( P^a \) to a low dimension vector. We firstly use a two-layer BN-ReLU-Conv [19] structure with \( H \) filters to introduce contextual information, and then adopt the average global pooling method [28] to obtain the final matching pattern-based similarity vector \( v_a \in \mathbb{R}^H \).

### 3.4 Aggregation Module

The similarity vectors from previous two modules are combined to compute the question similarity score \( y^q \) in this module. Given \( v_q \) and \( v_a \), we introduce a gate mechanism inspired by GRU [10], which takes the former as the primary one and the latter as the complementary one, to obtain the final question similarity vector \( v \). Specifically, it can be computed by

\[ r = \sigma(W_r v_a + U_r v_q), \]
\[ z = \sigma(W_z v_a + U_z v_q), \]
\[ \tilde{v} = \tanh(W_v r + U(r \otimes v_q)), \]
\[ v = zv_q + (1 - z)\tilde{v}, \]

where \( \otimes \) is the element-wise multiplication, \( \sigma \) denotes the sigmoid function, and \( W_r, W_z, W_v, U_r, U_z, U \) are trainable parameters.

Based on the question similarity vector \( v \), we then apply a multi-layer perceptron (MLP) to obtain the question similarity score \( y^q \),

\[ y^q = \sigma(W_2 \text{ReLU}(W_1 v + b_1) + b_2), \tag{2} \]

in which \( W_1, b_1, W_2 \) and \( b_2 \) are trainable parameters.

### 3.5 Model Training and Inference

In the training phase, we employ the cross-entropy loss to learn our \( \text{Match}^3 \) model in an end-to-end way. To train the model sufficiently, we adopt a multi-task learning strategy to combine the question-question matching task and the question-answer matching task. The question-question matching task aims to measure the similarity between two questions as our main task, while the question-answer matching is an auxiliary task that aims to evaluate whether the answer can satisfy the question in the matching pattern-based similarity module. For the auxiliary task, we employ the \( B^p \) (see Eq. 1) from the matching pattern layer for prediction. We apply a multi-layer perceptron described in Equ. 2 to calculate the similarity score \( y^a \) between the user question and the archived answer. In the same way, we get \( y^a \) to represent the archived question answer pair similarity score. However, due to the lack of question-answer matching labels, we should build the ground-truth for the auxiliary task. In details, (1) for each archived question, we regard the corresponding archived answer as the relevant answer; (2) for each user question, we regard the corresponding answer with respect to its similar question as the relevant answer. Thus, we computed the question-answer matching loss \( \text{loss}^q \) and \( \text{loss}^a \) with cross-entropy loss again. The overall loss is defined as the weighted sums of three losses, i.e.,

\[ \text{loss} = r\text{loss}^q + \frac{1 - r}{2}\text{loss}^u + \frac{1 - r}{2}\text{loss}^a. \]
Table 2: Dataset statistics. \# denotes the number of instances, \(|\text{len}_Q|\) and \(|\text{len}_A|\) denote the average length of the questions and answers, respectively.

| Dataset          | \#Train | \#Dev | \#Test | \(|\text{len}_Q|\) | \(|\text{len}_A|\) |
|------------------|---------|-------|--------|----------------|----------------|
| CQADupStack      | 56,633  | 5000  | 5000   | 11.89          | 177.70         |
| QuoraQP-a        | 281,480 | 10,000| 10,000 | 13.83          | 45.65          |

where \(r \in [0, 1]\) is the main task loss ratio. To overcome the issue of sparse irrelevant answers, for each question, we random sample irrelevant answers from its top-\(K^3\) candidate answers which are retrieved from the whole answer collection by BM25 \[34\] method. Note if the answer is irrelevant to both the user question and the archived question, we set \(r = 0\) while training this instance because the answer can not be a bridge in this situation.

In the inference phase, given the user question \(Q^u\), the archived question \(Q^a\) and the real archived answer \(A^a\), we compare the prediction \(y^g\) with the threshold 0.5 to identify whether the questions are similar or not.

4 EXPERIMENTS

4.1 Datasets

We evaluate our model on the following two datasets, i.e., CQADupStack and QuoraQP-a (answer-expanded version of QuoraQP). The detailed statistics of these datasets are shown in Table 2.

- **CQADupStack** is a benchmark dataset which is widely used in CQA \[21\]. It contains question threads sampled from twelve StackExchange subforums and annotated with similar question information. We take the annotated best answer of the question as the archived answer. If there is no best answer for the question, we directly use the answer with the highest score as the archived answer.

- **QuoraQP-a** is built on the widely used CQA dataset QuoraQP \[6\], which contains 537,933 distinct questions from Quora. The original dataset cannot be used for our task directly since it does not include archived answers. To evaluate our model, we randomly select one question in each pair as the user question and another one as the archived question. Then, we take the top ranked answer from the original website\(^7\) as the archived answer\(^8\).

4.2 Baseline Models

We compare our proposed model with previous similar question identification methods, which could be classified into two categories based on the usage of answers, i.e., question-only methods and one-side methods.

4.2.1 Question-only Methods. Here we consider six existing methods which only rely on questions for similar question identification.

- **TSUBAKI** \[37\] accounts for a dependency structure of a sentence and synonyms to evaluate the question similarity.

- **BiMPM** \[43\] employs a bilateral mechanism to enhance single direction matching in sentence pair relevance modeling.

- **ESIM** \[9\] is a sequential inference model based on chain LSTMs, which considers the recursive architectures in both local inference modeling and inference composition.

- **DIIN** \[15\] is a instance of Interactive Inference Network (IIN) architecture that hierarchically extracting semantic features from interaction space.

- **RE2** \[48\] is a fast and strong neural model with stacked alignment layers, which also employ fusion layer to make the model deeper.

- **Bert** \[11\] is a pre-trained language model based on stacked Transformer \[39\] layers, which is effective in measuring the text pair similarity.

4.2.2 One-side Methods. We also consider recently proposed methods that employ one-side usage of the archived answer for similar question identification.

- **TSUBAKI-Bert** \[36\] is a recently proposed method that combine the similarity between questions and the relevance between the user question and archived answer.

Here, to fully demonstrate the effectiveness of our model, we also incorporate answers into those question-only methods by two basic operators \[16\]. The first one is to directly concatenate the archived question along with its answer. We denote these methods as M\(_{\text{concat}}\), where M could be any method in the question-only Methods. The second one is M\(_{\text{atttn}}\), which effectively combines the similarity representations from both the question pair matching and the question answer matching using attention mechanisms in a hierarchical manner.

4.3 Implementation Details

We implement our model by Tensorflow \[1\]. The hyper-parameters are tuned with the development set. The model is trained end-to-end by RAdam \[29\] optimizer. We set the learning rate of RAdam as \(5e - 5\) and other parameters as \(\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e - 6\). We use 3 \(\times\) 3 kernels in the compression layer. We use an exponential decayed keep rate during training, where the initial keep rate is 1.0 and the decay rate is 0.933 for every 5000 steps, where the keep rate will achieve to 0.5 after 50,000 steps. We initial the Bert structure in our model with released Bert-base model\(^9\). The other parameters are randomly initialized under a normal distribution with \(\mu = 0\) and \(\sigma = 0.2\). The maximum question length is truncated to 24 for CQADupStack and 32 for QuoraQP-a. The maximum answer length is truncated to 256 for CQADupStack and 100 for QuoraQP-a. The batch size is 32 for CQADupStack and 48 for QuoraQP-a.

For evaluation, we adopt Accuracy, Precision, Recall and F1 score to evaluate the models, and set the Accuracy as the main metric.

4.4 Hyper-parameter Analysis

4.4.1 Pattern Similarity Function. As described in Section 3.3.2, we can adopt various pattern similarity functions to calculate the pattern similarity tensor. Here, we study the performance of five candidate functions. The results are shown in Table 3. As we can see, the choice of pattern similarity function does affect the performance of the Match\(^5\) model. Specifically, the dot function has

\(^5\)We set \(K = 5\) in this paper.

\(^6\)https://www.kaggle.com/c/quora-question-pairs

\(^7\)https://www.quora.com/

\(^8\)We released the dataset at http://tinyurl.com/y8kbbfyu

\(^9\)https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip
Table 3: Results of different similarity functions in the matching pattern-based module on CQADupStack.

<table>
<thead>
<tr>
<th>Function</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>dot</td>
<td>62.84</td>
<td>56.34</td>
<td>55.12</td>
<td>55.72</td>
</tr>
<tr>
<td>cos</td>
<td>62.80</td>
<td>56.29</td>
<td>55.07</td>
<td>55.67</td>
</tr>
<tr>
<td>l1</td>
<td>62.44</td>
<td>55.90</td>
<td>54.31</td>
<td>55.09</td>
</tr>
<tr>
<td>l2</td>
<td>62.46</td>
<td>55.89</td>
<td>54.55</td>
<td>55.21</td>
</tr>
<tr>
<td>jss</td>
<td>62.76</td>
<td>56.25</td>
<td>54.97</td>
<td>55.60</td>
</tr>
</tbody>
</table>

![Figure 4: Results of different main task loss ratios of Match² on CQADupStack.](image)

The number of correct predictions in positive question pairs (b) The number of correct predictions in negative question pairs

Figure 4: Results of different main task loss ratios of Match² on CQADupStack.

4.4.2 Multi-task analysis. In the model learning phase, we introduced an additional task to train the model. The final optimization objective of the model is the linear combined loss with pre-defined main task loss ratio \( \gamma \). Here, we study how this ratio affects the model performance. Specifically, we set the weight value from 0.2 to 1.0, where the larger value denotes more emphasis on the main task, i.e., similar question identification task. The results are depicted in Figure 4, we can see that there is a consistent tendency between all the evaluation metrics, i.e., the performance first improves along with the increase of the weight value, and drops when the weight becomes larger than 0.6. The best performance can be obtained at 0.6, where the model pays balanced attention to both learning objectives.

4.5 Main Results

In this section, we show the main results of the Match² model as well as baseline methods. All the results are summarized in Table 4.

Firstly, for the question-only methods, we can see that neural models (e.g., BiMPM, ESIM and etc.) achieve significant better performance than traditional methods (i.e., TSUBAKI) on both datasets. Moreover, it can be observed that the relative improvement of the neural methods over TSUBAKI is much larger on QuoraQP-a than the CQADupStack. The reason might that the QuoraQP-a is much larger in size than the CQADupStack, where neural models are often data hungry. The Bert achieves the best performance on both datasets in terms of all metrics. This indicates the models pretrained on a large amounts of unstructured texts learn to encode linguistic features that improve the performance.

Secondly, comparing the one-side methods with the question-only methods, we can find that incorporating the answers could indeed improve the performance. However, there are also some methods achieving inferior performance with the archived answer. For example, the accuracy of RE\_conc is decreases from 60.56 to 60.16 on the CQADupStack. This demonstrates that simply incorporating the answers could introduce unexpected noises, which could possibly hurt the performance. Moreover, we find that the attention method is relatively more effective than the concatenation method, which indicates the possibility to improve the performance by carefully designed answer usage method.

Thirdly, the Match² model achieves the best performance in terms of all metrics on both benchmarks. For example, the relative improvement of the Match² model over the best performing baseline method (i.e., Bert\_att) is about 3.3% and 1.3% in terms of F1 metric on CQADupStack and QuoraQP-a. All these demonstrate the effectiveness of the Match².

4.6 Analysis on the Match²

To better analyze the effect of different components in Match², we first construct three variants of the model, then evaluate them on both benchmarks and on different question groups. The constructed variants are listed as follows:

- Match\_Q² is used to represent the representation-based module. It removes the matching pattern-based module and use a multilayer perceptron (MLP) to replace the aggregation module.
Table 4: Main Results on CQADupStack and QuoraQP-a. †indicates the statistically significant difference over the best baseline model, where +/- indicates the statistically significant improvement/deterioration over the question-only counterpart with $p < 0.01$.[49]

<table>
<thead>
<tr>
<th>Method</th>
<th>#</th>
<th>Model</th>
<th>CQADupStack</th>
<th>QuoraQP-a</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accuracy</td>
<td>Precision</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>TSUBAKI</td>
<td>56.20</td>
<td>50.17</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>BiMPM</td>
<td>59.44</td>
<td>54.84</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>ESIM</td>
<td>58.64</td>
<td>53.85</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>DIIN</td>
<td>60.30</td>
<td>56.13</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>RE2</td>
<td>60.56</td>
<td>56.47</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Bert</td>
<td>60.92</td>
<td>56.91</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>TSUBAKI+Bert</td>
<td>57.20+</td>
<td>51.75+</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>BiMPM(_{concat})</td>
<td>59.48</td>
<td>54.55</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>ESIM(_{concat})</td>
<td>59.05+</td>
<td>54.35</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>DIIN(_{concat})</td>
<td>60.74+</td>
<td>56.31+</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>RE2(_{concat})</td>
<td>60.16−</td>
<td>54.96−</td>
</tr>
<tr>
<td></td>
<td>one-side</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>Bert(_{concat})</td>
<td>61.50+</td>
<td>56.70</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>BiMPM(_{attn})</td>
<td>59.74</td>
<td>55.11</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>ESIM(_{attn})</td>
<td>59.38+</td>
<td>54.93+</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>DIIN(_{attn})</td>
<td>60.78+</td>
<td>56.84+</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>RE2(_{attn})</td>
<td>61.18+</td>
<td>56.55</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>Bert(_{attn})</td>
<td>61.96+</td>
<td>57.31+</td>
</tr>
<tr>
<td></td>
<td>two-side</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>Match(_{2})</td>
<td>62.78†</td>
<td>58.02†</td>
</tr>
</tbody>
</table>

Table 5: Ablation results on CQADupStack and QuoraQP-a. †indicates the statistically significant difference over the Match\(_{2}\) model with $p < 0.01$.[49]

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match(_{2})</td>
<td>62.78</td>
<td>58.02</td>
<td>55.03</td>
<td>56.49</td>
<td>90.65</td>
<td>86.21</td>
<td>86.29</td>
<td>86.25</td>
</tr>
<tr>
<td>Match(_{2Q})</td>
<td>60.94†</td>
<td>56.92†</td>
<td>45.33†</td>
<td>50.47†</td>
<td>89.24†</td>
<td>84.21†</td>
<td>84.11†</td>
<td>84.16†</td>
</tr>
<tr>
<td>Match(_{2A})</td>
<td>60.44†</td>
<td>55.46†</td>
<td>50.25†</td>
<td>52.72†</td>
<td>89.11†</td>
<td>83.32†</td>
<td>84.97†</td>
<td>84.14†</td>
</tr>
<tr>
<td>Match(_{2_{attn}})</td>
<td>62.32†</td>
<td>58.00</td>
<td>51.34†</td>
<td>54.47†</td>
<td>90.04†</td>
<td>85.42†</td>
<td>85.26†</td>
<td>85.34†</td>
</tr>
</tbody>
</table>

- **Match\(_{2A}\)** is for matching pattern-based module. It removes the representation-based module and use a MLP for aggregation.
- **Match\(_{2\_{attn}}\)** adopts the attention mechanism [16] to replace the gate mechanism in the aggregation module, to analyze the effect of the gate mechanism.

4.6.1 Sub-Module Analysis. The performance of different variants are shown in Table 5. Firstly, we can see that both of the Match\(_{2Q}\) and Match\(_{2A}\) achieve relatively good performance with the sub-module itself, which demonstrates these modules are effective in most cases. Secondly, comparing with these two variants, we find that Match\(_{2Q}\) achieves higher precision while Match\(_{2A}\) achieves higher recall. This indicates the representation-based module and the matching-pattern based module could be complementary to each other. Finally, we observe the attention mechanism cannot fully utilize the advantages of the previous modules, which is particularly reflected in the recall metric. This difference demonstrates the effectiveness of the gating mechanism in the aggregation component.

4.6.2 Question Group-based Analysis. For more detailed analysis of model performance, we divide the question pairs in CQADupStack into twenty groups based on their similarity and Jaccard Index [23], which is a widely-used word level similarity feature. We analyze the number of correct predictions in each group. The results are shown in Figure 5.

We notice that the positive and negative question pairs have similar Jaccard Index distribution in CQADupStack. Specifically, for the positive question pairs, we can see the Match\(_{2Q}\) achieves better performance than the Match\(_{2}\) on all the groups, and the gap is larger on the pairs with higher Jaccard Index, i.e. more shared words. It indicates that the Match\(_{2\_{attn}}\) could directly capture the word similarities, which is useful to the similar questions with many shared words.
Table 6: Two cases from the CQADupStack data. Match$_Q^2$ is the representation-based similarity module, and Match$_A^2$ is the matching pattern-based similarity module.

<table>
<thead>
<tr>
<th>Case</th>
<th>Ground-truth</th>
<th>Match$_Q^2$</th>
<th>Match$_A^2$</th>
<th>Match$_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case I</td>
<td>$Q^2$: how to keep a session when logging out $Q^2$: can I close a terminal without killing the command running in it $A^2$: once you log out a terminal, this kill the running session in it as well. to keep the session alive, you should start a session with 'nohup' command. another way is pause the session with 'ctrl-z', pull it into the background with 'bg' and then 'disown' it.</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Case II</td>
<td>$Q^2$: how do I close a terminal</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6: Two cases from the CQADupStack data. Match$_Q^2$ is the representation-based similarity module, and Match$_A^2$ is the matching pattern-based similarity module.

On the other hand, for the negative question pairs, we can observe that the Match$_Q^2$ could not well address the negative questions pairs with higher Jaccard Index. For example, when the Jaccard Index higher than 0.8, the Match$_Q^2$ fails on all the instances. The Match$_A^2$ outperforms the Match$_Q^2$ especially on the higher Jaccard Index groups. This demonstrates that the matching pattern could avoid the noises from shared words and emphasize the difference between questions. Finally, the Match$_A^2$ module could outperform these two types of modules in most cases. It indicates the effectiveness of the gate mechanism that combines the advantage of these two module into a unified model.

4.7 Case Study and Visualization

Here, we conduct case studies to better understand what have been learned by the Match$_A^2$ model. We also take the Match$_Q^2$ and Match$_A^2$ for comparison. The instances are shown in Table 6, the first one is a positive question pair with few shared words, while the second one is a negative pair with more common words. We can see that the Match$_Q^2$ is not good at dealing with these types of questions, but the Match$_A^2$ could correctly identify them with the help of Match$_A^2$. Specifically, we visualize the matching patterns and pattern similarity tensors from Match$_Q^2$ and Match$_A^2$ in Figure 6. For case I, we notice the Match$_Q^2$ is difficult to find out the semantic relation between questions but only recognizes the cluttered similarity presented in Figure 6(a). By leveraging the archived answer as a bridge, the Match$_A^2$ can easily identify the similarities by comparing the matching patterns, as shown in Figure 6(b).

For case II, we notice that the Match$_Q^2$ highlights three similar phrases in Figure 6(c), and makes a false positive prediction that the questions are similar. On the other hand, as shown in Figure 6(d), the only semantic relevance between two questions is “how” which means the questions are different except their question type. Based on the pattern similarity tensor, the Match$_A^2$ is able to predict these two questions as different.

5 CONCLUSION AND FUTURE WORK

In this paper, we introduced a two-side usage of the archived answer for similar question identification task by leveraging the answer as a bridge of the questions. We proposed a novel matching over matching (Match$_A^2$) model, which consists of three main components, namely the representation-based similarity module, matching pattern-based similarity module, and the aggregation module. Empirical experiments on two benchmarks demonstrate that our model can significantly outperform previous state-of-the-art methods. Moreover, we also conducted rigorous experiments on the sub-modules to verify the effectiveness of the model. In the future work, we would like to extend our model to leverage variant number of answers and take the answer quality into account.