MIX: Multi-Channel Information Crossing for Text Matching

Haolan Chen
Mobile Internet Group, Tencent
haolan@ualberta.ca

Fred X. Han
University of Alberta
xuefei1@ualberta.ca

Di Niu
University of Alberta
dniu@ualberta.ca

Dong Liu
Mobile Internet Group, Tencent
dougliu@tencent.com

Kunfeng Lai
Mobile Internet Group, Tencent
calvinlai@tencent.com

Chenglin Wu
Mobile Internet Group, Tencent
ether.wcl@gmail.com

Yu Xu
Mobile Internet Group, Tencent
henrysxu@tencent.com

ABSTRACT

Short Text Matching plays an important role in many natural language processing tasks such as information retrieval, question answering, and dialogue systems. Conventional text matching methods rely on predefined templates and rules. However, for a short piece of text with a limited number of words, these rules are unable to generalize well to unobserved data. With the success of deep learning in fields like computer vision, speech recognition and recommender systems, many recent efforts have been made to apply deep neural network models to natural language processing tasks to reduce the cost of manual feature engineering. In this paper, we present the design of Multi-Channel Information Crossing (MIX), a multi-channel convolutional neural network (CNN) model for text matching in a production environment, with additional attention mechanisms on sentences and semantic features. MIX compares text snippets at varied granularities to form a series of multi-channel similarity matrices, which are then crossed with another set of carefully designed attention matrices to expose the rich structures of sentences to deep neural networks. We implemented MIX and deployed the system on Tencent's Venus distributed computation platform. Thanks to the well-engineered multi-channel information crossing, evaluation results suggest that MIX outperforms a wide range of state-of-the-art deep neural network models by at least 11.1% in terms of the normalized discounted cumulative gain (NDCG@3), on the English WikiQA dataset. Moreover, we performed online A/B tests with real users on the search service of Tencent QQ Browser. Results show that MIX raised the number of clicks on the returned results by 5.7%, due to the increased accuracy in query-document matching, which demonstrates the superior performance of MIX in a real-world production environment.

CCS CONCEPTS

• **Information systems → Information retrieval**; **Learning to rank**; **Question answering**; • **Computing methodologies → Neural networks**;

KEYWORDS

Text matching, ad-hoc retrieval, question answering, convolutional neural networks, attention mechanism.

ACM Reference Format:
Haolan Chen, Fred X. Han, Di Niu, Dong Liu, Kunfeng Lai, Chenglin Wu, and Yu Xu. 2018. MIX: Multi-Channel Information Crossing for Text Matching. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3219928

1 INTRODUCTION

Short Text Matching plays a critical role in many natural language processing tasks, such as information retrieval, question answering, and dialogue systems. Early methods for text matching include automatic question answering by retrieving a knowledge base, and adhoc retrieval based on word matching and feature crossing [17, 24]. However, these methods all depend on manually defined templates and rules, which limits the generalizability of a well tuned model and its portability toward different task requirements. Recent advancements in deep neural network models have brought about new opportunities to enhance the natural language processing performance. By alleviating the need of manual feature engineering, deep network models can better generalize to a variety of tasks. In recent years, a number of deep network architectures have been proposed for short text matching based on convolutional neural networks and recurrent neural networks [2, 3, 6, 7, 9, 11–15, 19–21, 23].

In this paper, we perform a reality check of a wide range of recent deep learning techniques for text matching We point out that despite the novelties in the respective deep network models, there still exists significant room for performance improvements when these methods are put into practice, especially when deep models are combined with the analysis of linguistic structures and semantic features. In particular, we present our design of Multi-Channel Information Crossing (MIX), a multi-channel convolutional neural network (CNN) model for text matching that achieves superior performance in the production environment at Tencent.

MIX is a novel fusion of CNNs at multiple granularities along with carefully designed attention mechanisms. The general ideas behind MIX can be summarized as follow: *First*, MIX represents text snippets with features extracted at multiple granularities related to terms, phrases, syntax and semantics, term frequency and weights, and even grammar information, which we observe from our experiments, to be a necessary practice to fully realize the potentials of deep models. The combination of text matching on multiple levels of features will maximize the ability of deep architectures to express all levels of local dependences and minimize information loss during convolution. *Second*, MIX also presents a novel fusion technique to combine the matching results obtained from multiple channels. There are two types of channels in MIX, through which features of the two text snippets can interact. The first type is a semantic information channel, which represents the meaning of text such as unigrams, bigrams and trigrams. The second type of channel contains structural information such as term weights, Part-Of-Speech and Named Entities as well as the spatial relevance of the interactions. In MIX, semantic information channels play the role of similarity matching, while structural information channels are used as attention mechanisms. Moreover, MIX uses 3D convolution kernels to process these stacked layers, extract abstract features from multiple channels and combine outputs through a multilayer perceptron [5]. The channel combining mechanism allows MIX to easily incorporate new channels into its learning framework, enabling MIX to be applicable to a wide range of tasks.

We implemented and deployed MIX on Tencent’s Venus distributed processing platform. We evaluated MIX based on multiple datasets as well as via online A/B tests in the traffic of Tencent QQ mobile browser, which has the largest market share in Chinese mobile browser market. In the offline evaluation part, we tested MIX on an English Question Answering dataset WikiQA[25] and a Chinese search result dataset collected from the QQ mobile browser. WikiQA is a publicly accessible dataset containing open-domain question and answer pairs provided by Microsoft. On the WikiQA dataset, MIX outperforms a wide range of state-of-the-art methods by at least 11.1% on NDCG@3, which is a popular metric for measuring the ranking quality and is widely adopted in search engine evaluation. The other Chinese search result dataset is collected from Tencent QQ Browser with user consents and is sampled from the online search traffic produced by 10 million active users per day. The dataset includes 120,000 query-documents entries and reviewer-generated labels, which indicate the degrees of matching for every query-document pair in the dataset. On this dataset, MIX outperforms all other state-of-the-art methods by at least 8.2% in terms of NDCG@3.

Moreover, during the online A/B testing in Tencent QQ mobile browser, MIX led to a 5.7% increase in the click through rate (CTR) as compared to traffic without MIX. Evaluation results demonstrate the superior capability of MIX in improving text matching accuracy in a production environment, as well as its ability to generalize across datasets in different languages.

2 PRELIMINARIES AND BACKGROUND

Depending on the order of transformation and matching, text matching models can be divided into two categories: Presentation based and Interaction based. The former first transforms every piece of text to a tensor representation with deep neural networks, such as Deep Semantic Similarity Model(DSSM) [8], Convolutional Deep Semantic Similarity Model(CDSSM) [18], Multiple Granularity CNN(Multigrancnn) [26], Convolutional Neural Tensor Network(CNTN) [16] etc. Matching between two text instances is then performed on their vector representations. Conversely, latter methods first generate an interaction matrix for every text pair, then utilize neural networks to extract useful features and learn meaningful matching patterns from the interaction matrices, such as Arc-I [7], MatchPyramid [15] and DRMM [6].

2.1 Presentation based Methods

Presentation based methods generate distributed representation from text inputs through deep neural networks. There are a number of work employing this method, including CNN based [11, 12], RNN based [13, 14] and tree-based RNN methods [9, 20]. They are inspired by the Siamese network structure [2], in which the vector representation of a text input is first generated, then the degree of matching is computed as the Euclidean distance between two vectors. They differ mainly in the procedure to construct the representations and the way of calculating a matching degree.

Huang et al. proposed DSSM [8]. DSSM is the earliest research effort to apply a deep neural network model on text matching. In DSSM, each piece of text is vectorized through a 5-layer neural network and then a matching score for a text pair is calculated as the cosine similarity between their representations. The model first splits sentences into a list of tri-letters, and hashes the bag of tri-letters into a vector. The tri-letter layers are fed into a 3-layer MLP to produce a semantic vector for the whole sentence. Compared to traditional text matching models, DSSM shows significant improvements on the NDCG [22] metric.

Following DSSM, Shen et al. argues that the MLP in DSSM contains too many parameters which increases the overall complexity of the model and is in greater risk of over-fitting. The bag-of-tri-letter model also ignores word positions, which are important features in text matching. Therefore, Shen et al. proposes Convolutional DSSM [18], which takes in sequences of words as input. Compared with DSSM, CDSSM replaces the MLP with CNN when generating text representations. The precision of matching is further improved in CDSSM.

Qiu et al. proposes CNTN, which also utilizes CNN to represent the semantic meaning of text inputs. However instead of Euclidean distance or MLP interaction, CNTN matches text representations with a tensor, thus better characterizing the complicated relations between two representations. Tensor neural network is proposed by Socher et al [19]. CNTN shows great performance in community Question Answering tasks.

2.2 Interaction based Methods

Compared with presentation based methods, Interaction based methods aim to capture direct matching features: the degree and the structure of matching. This kind of methods are more intuitive and comply to the nature of language matching problems. Given two pieces of text, matching of key words are considered in the first place, then their relative positions are also taken into account. At last, both features are combined to compute the final matching degree. Recent works show that this type of methods perform better in multiple text matching tasks.

Hu et al. proposed ARC-II [7], which first represents a sentence as a sequence of word vectors, and then adjusts the sliding windows in the first convolution layer to focus on adjacent word vectors. The first convolution layer outputs a 3-d tensor as the representation of relations between two sentences. Afterwards, multiple convolution-pooling operations are performed on the 3-d tensor. In the end, a high-level abstract representation describing the association between two sentences are obtained. A MLP is employed to consolidate this abstraction into a single matching degree. Compared with DSSM, ARC-II considers the order of words in a sentence, thus it is able to better describe the association between general matching and element-wise matching. In terms of evaluation results, ARC-II also shows better performance than DSSM and CDSSM.

MatchPyramid is proposed by Pang et al [15]. The paper formalizes the interaction between two pieces of text as a Matching Matrix. On the 2-d matrix, convolution is performed to extract the pattern of interactions. The matching score is calculated by a fully connected network. MatchPyramid defines the Matching Matrix as a matrix of matching degrees between the words of two sentences. This model computes each pair-wise matching degree using cosine similarity. In essence, the Matching Matrix can be thought as a gray scale image. Similar to an image classification task, multiple layers of convolution and pooling is then executed on this "image".

DRMM is proposed by Guo et al [6]. When most NLP tasks focus on semantic matching, the Ad-hoc retrieval task is mainly about relevance matching, i.e., identifying whether a document is relevant to a given query. DRMM is an interaction-focused model which employs a joint deep architecture at the query terms vs. document terms level. Specifically, DRMM first builds local interactions between each pair of terms from a query and a document based on term embeddings. For each query term, the model map the variable-length local interactions into a fixed-length matching histogram. From there, a feedforward matching network is employed to learn hierarchical matching patterns and produce a matching score. Finally, the overall matching score is generated by aggregating the scores from each query term with a term gating network computing the aggregation weights.

The recent KNRM [23] and Conv-KNRM [3] directly makes interaction between ngrams' embeddings from two pieces of text and employs a kernel pooling layer to combine the cross-match layers to generate the matching score.

Interaction based methods consider the matching degree and matching structure at the same time, they achieve significant improvements in multiple text matching tasks. However these approaches often suffer from the following weaknesses:

(1) Words or n-grams are regarded as the basic semantic units, which ignores many other useful aspects of natural languages, such as syntactic information or cross references between sentences.
(2) They can hardly well describe the relation between global matching and local matching. In reality, matching of critical parts or certain patterns in a text pair is often more important than matching of global structures.
(3) The lack of a unified ensemble mechanism for multiple aspect matching. It is difficult for the model to extend to new tasks by adding only new matching features.

Overall, Many of the aforementioned models rely too much on the generalization ability of neural networks, as well as the quality of the training data.

3 MIX MODEL

In this section, We describe the detailed model adopted in multi-channel information crossing (MIX) for boosting text matching performance. We define *global matching* as the matching between two sentences and *local matching* as the matching between text elements within sentences. Inspired by interaction-based models, MIX models the relevance between two pieces of text through a combined use of global matching and local matching techniques. Relying on the strong representational learning abilities of deep neural networks, MIX is capable of hierarchically and multi-dimensionally depicting the essence of a text matching problem. As shown in Fig. 1, MIX effectively divides the text matching problem into the following subproblems:

First, as we can see in the upper-left part of Fig. 1, sentences are parsed into text snippets at various granularities, such as *unigrams*, *bigrams* and *trigrams*. This way, MIX improves the accuracy of local matching by finding the most appropriate semantic representations for a piece of text, which could be either words, terms or phrases. The goal here is to capture of the most information at different levels of interaction.

Second, as shown in the *attention units* part of Fig. 1, we extract grammar information such as relative weights and Part-of-Speech (PoS) tags, from which we design attention matrices in the *attention channels* to encapsulate the rich structural patterns. With this method, we first investigate the relationships between global matching and local matching. Then, we justify how our attention mechanisms constitute a way to construct global matching on top of local matching to strengthen the overall matching quality.

Third, as illustrated in *weighed channels* and the *2D-convolution* parts of Fig. 1, we cross *locally matched channels* and *attention channels* to extract significant feature combinations for local matching. We describe the mechanisms of MIX in detail in the following subsections.

3.1 Local Matching

In many related works, text matching is achieved by matching word embedding vectors at different levels [6, 15] These state-of-the-art models construct hierarchical neural networks to extract patterns of matching at term-level or phrase-level. However, using word embedding vectors as the only form of semantic representation results in information loss and undermines the intuition of higher-level matching.

As shown in Fig. 2, local matching based on word embedding vectors has two limitations. On one hand, In Fig. 2(a), phrase *senic spot* and phrase *place of interest* are semantically similar. However,

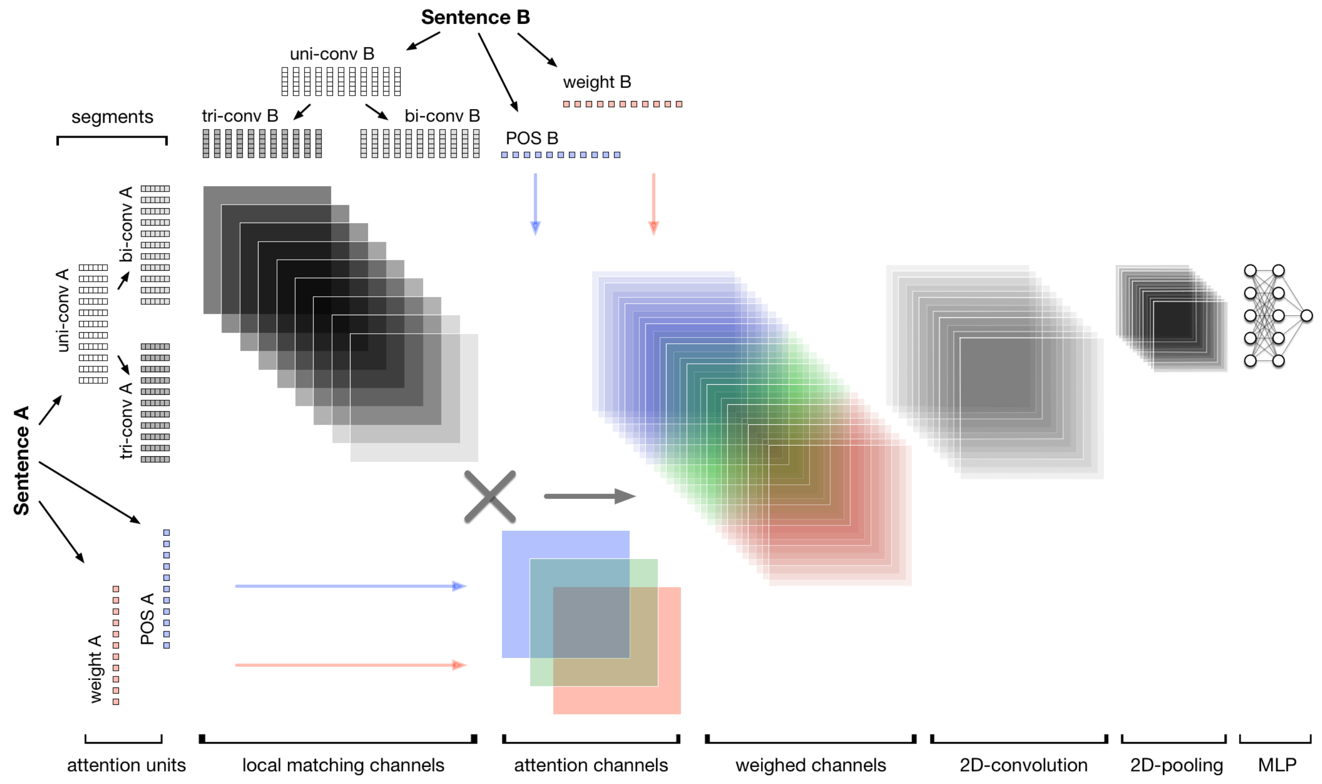
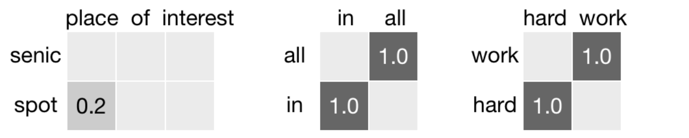


Figure 1: Architecture of the Multi-channel Information Crossing Model.



(a) Phrases of different words share the same meaning (b) Phrases composed of same words differs in meaning (c) Phrases composed of same words can be the same

Figure 2: Case study of word matching shows patterns of matching are not able to preserve original information.

similarity between their word embedding vectors is low, which has an undesirable negative effect on the overall matching result. On the other hand, phrase *all in* and phrase *in all* consist of the exact same set of words. However, they differ completely in meaning as shown in Fig. 2(b). With these weaknesses in the input representation, even a strong learner like a deep neural network will have difficulties extracting an accurate matching pattern. Consider another example shown in Fig. 2(c), the matching matrix of *hard work* and *work hard* has exactly the same pattern as matrix of *all in* and *in all*. However, the former two pharses have extremely similar meanings while the latter pair entirely differs. This case study shows that text matching performed solely on the basis of word embedding vectors is largely ineffective. In other words, text matching from only one perspective cannot produce satisfying results.

To improve upon this single-perspective matching, we propose that a piece of text should be first split into a sequence of multigrams.

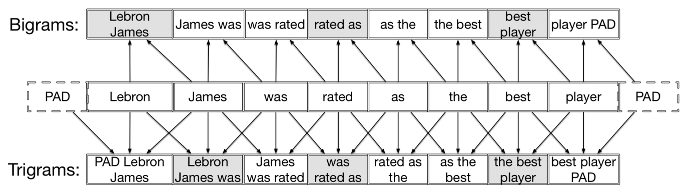


Figure 3: Case study of bigram convolution and trigram convolution, gray color indicates more possibility to be a phrase.

We choose a word from the text as the kernel word. Then, we use a fixed-length window to combine its surrounding words. In English, a minimal semantic unit usually contains 1 to 3 words, so we use a convolution window of size 1 to 3 to express the integral meaning of the phrase. For text$_1$ and text$_2$, we apply filters of size m and n respectively, where $m \in \{1, 2, 3\}, n \in \{1, 2, 3\}$. As shown in Fig. 3, we take 2-gram convolution and 3-gram convolution as examples, and we formalize the convolution as equation (1). N-grams that match existing English phrases will have higher weights in the training phase.

$$\mathbf{z}_i^{(1)}(\mathbf{x}) = f\left(\mathbf{W}^{(1)}\mathbf{z}_i^{(0)} + \mathbf{b}^{(1)}\right), \tag{1}$$

where $\mathbf{z}_i^{(0)}(\mathbf{x})$ stands for the unigram embeddings for location i in embedding layer; $\mathbf{z}_i^{(1)}(\mathbf{x})$ gives the convolution results for location i; $\mathbf{W}^{(0)}$ is the set of parameters of the convolution kernel functions,

$f(\cdot)$ is the activation function Relu, and $\mathbf{z}_i^0$ denotes the segment of embedding layer for the convolution at location i, while

$$\mathbf{z}_i^{(0)} = \mathbf{X}_{i:i+m-1} = \left[\mathbf{X}_i^T, \mathbf{X}_{i+1}^T, \ldots, \mathbf{X}_{i+m-1}^T\right]^T, \tag{2}$$

in which m is the size of the sliding window, which concatenates the vectors for words of filter size from sentence input X.

In the beginning of the training phase, it is hard to choose an appropriate size for the window in each convolution filter, therefore we preserve interactions of all the possible pairs of filter sizes. And we expect the following training phase to learn the appropriate weights in different channels.

For convenience purposes, we name the interaction channels between m-gram sequences of text_1 and n-gram sequences of text_2 as $\text{Layer}_{m,n}$.

3.2 Local and Global Matching

Under our assumptions of text matching, global matching largely depends on combinations of local matching. However, in reality local matching results would not equally influence the result of global matching. To capture this difference in weighting, we introduce attention mechanisms to the global matching procedure. We initialize the attention matrices with the following features:

First, the weighting of different terms in the interaction matrix should reflect the importance of each matching signal. In other words, matching of two key phrases has much more significance than matching of two trivial phrases when considering two sentences. We take the Question-Answer problem in Fig. 4 as an example. In Fig. 4(a), answer *Steve Curry won his first MVP in 2014* looks quite similar to question *What year did Lebron James win his first MVP*. They match exactly in multiple words. This would be a good match from the perspective of word embedding based matching, however it is incorrect in reality. As we can observe, the inaccuracy is caused by the lack of keywords weighting. The pair matched well in words *win, his, first, MVP, etc*, but the mismatch of importance subject *Lebron James* has not been taken into consideration.

We design an element-wise attention layer to measure the importance of each interaction. As shown in Fig. 4(b), the attention layer works like a mask on top of the original matching matrix, and it is initialized with the product of the term inverse document frequency (IDF) in the interaction. Weights of trivial term interactions such as *his-his, first-first, year-2014* are apparently smaller than those of others. After stacking the attention layer upon original matching layer and conducting a dot product operation, we obtain a weighed matching matrix in Fig. 4(c), where the intensity of trivial interactions is effectively decreased.

In contrast, as shown in Fig. 5, when answer *Lebron James was rated as the best player in 2009* matches with question *What year did Lebron James win his first MVP*, although local matching signals are not that intense, the matching of key phrase *Lebron James* contributes more to the matching result. From Fig. 5(c) we can conclude that after applying attention mechanism shown in Fig. 5(b), the local matching of key phrases is largely enhanced thus improves the matching accuracy. Finally, weights in the attention layer are also trainable.

Second, Part-Of-Speech (POS) features are also good measurements of importances in term interactions. For instance, matching between two named entities (like two person's names) always plays a more important role than the matching between regular nouns and adjectives. Ideally, given abundant training data and a model that generalizes well, this characteristic can be learned automatically during training. However, in real-world applications we usually only have limited training data that may or may not capture the strong influence of POS features. Therefore, introducing some prior knowledge related to useful POS matching is essential.

As shown in Fig. 6, based on original matching result Fig. 6(a), we extract POS tags from two pieces of text and initialize this attention layer with interactions of POS tags in Fig. 6(b). As we can see, the following interactions of POS tags show higher importance: interaction between two PERSON tags, interaction between two Verbs (VB), as well as interaction between Wh-prnoun concerning time (WP_time) and cardinal numbers (CD). After applying the POS-tag attention channel to Fig. 6(a), the key local matching signals are emphasized in Fig 6(c).

Third, the positions of words in a sentence can also influence the relation between local matching and global matching. For certain tasks, there exists a pattern of spatial importance. Take Question Answering (QA) tasks for example, the starting parts of both questions and answers usually have more importance than remaining parts of the text. As shown in Fig. 7, after training with a spatial attention layer, this figure indicates that the importance of interaction indeed differs with spatial information. And incorporating spatial weights can help better capture the relation between global matching and local matching.

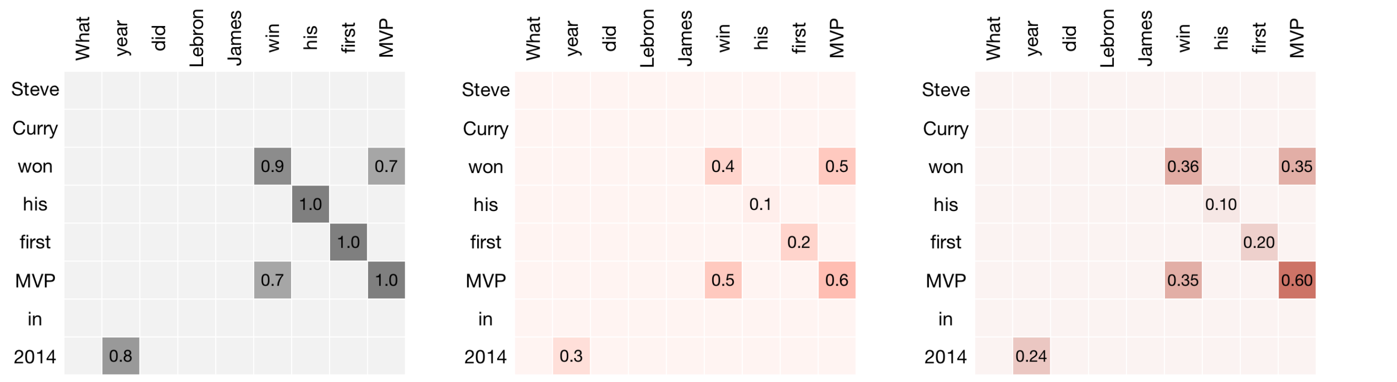
For convenience, we name the attention layer based on term weights as Att_{tw}, the attention layer based on POS features as Att_{pos}, and the layer based on word positions as $\text{Att}_{\text{spatial}}$. Moreover, we denote the weighting of a local matching channel $\text{Layer}_{m,n}$ with an attention layer Att_{ch} as $\text{Layer}_{m,n} \cdot \text{Att}_{\text{ch}}$.

We create permutations of attention layers as well as local matching layers. As shown in Fig. 1, we construct $\mathbf{3} \cdot \mathbf{M} \cdot \mathbf{N}$ layers in the form of: $\text{Att}_{\text{ch}} \cdot \text{Layer}_{m,n}$, $\text{ch} \in \{\text{tw}, \text{pos}, \text{spatial}\}$, $m \in \{1, \ldots, M\}$, $n \in \{1, \ldots, N\}$. By extracting information from multiple perspectives and constructing the attention layers, we effectively model the relationship between global matching and local matching. In this paper, we propose a pattern combining multiple channels of text interactions without the loss of generality.

3.3 Combination of Matching

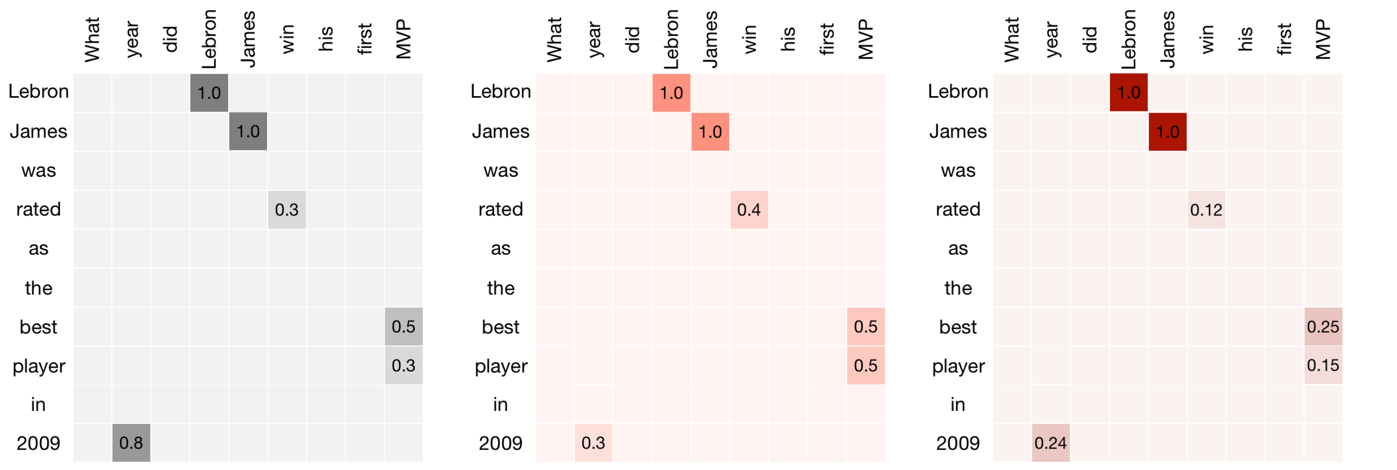
In 3.1, we construct multiple layers of local matching. In 3.2, we emphasize on the matching of key information by employing attention layers from three perspectives. However, the combination of weighed local matching has not yet been well modeled. To address this issue, we discuss this procedure in two aspects: first, the combination of different layers. As mentioned previously, each layer could be one of these types: $\{\text{Att}_{\text{ch}} \cdot \text{Layer}_{\text{m,n}}, \text{ch} \in (\text{tw}, \text{pos}, \text{spatial}), m \in \{1, \ldots, M\}, n \in \{1, \ldots, N\}\}$. Also, the combination of local matching information within a layer.

Although it may seem unlikely at first, both types of combinations can be modeled effectively with convolution. Because our goal here is to consolidate all the useful information into a final output



(a) Strong local matching leads to ill matching (b) Term weight channel as attention (c) Term weight attention decreases intensity of local matching

Figure 4: Case study shows strong local matching signals may lead to ill-matching. However, the term weighting channel acts as an attention mechanism to better depict the relations between global matching and local matching.



(a) Weak local matching signal leads to good matching (b) Term weight channel as attention (c) Term weight attention increases intensity of local matching

Figure 5: Case study shows weak local matching signal may also lead to good matching as the term weighting channel acts as an attention mechanism to better depict the relations between global matching and local matching.

matrix, we need a method that could iteratively discover and focus on one or few key interactions in a potentially large interaction matrix, or even a set of matrices from multiple layers. Convolution is the perfect candidate for this operation due to its sliding window mechanism. We formalize the inter-layer and intra-layer combination procedures as follow:

$$\mathbf{z}_{i,j}^{(1,k)} = f\left(\sum_{s=0}^{r_k-1}\sum_{t=0}^{r_k-1} \mathbf{W}_{s,t}^{(1,k)} \cdot \mathbf{z}_{i+s,j+t}^{(0)} + b^{(1,k)}\right), \tag{3}$$

where $\mathbf{z}_{i,j}^{(1,k)}$ gives output of the feature map; $\mathbf{W}^{(1,k)}$ is the parameters of convolution kernel functions from kernel 1 to kernel k; $f(\cdot)$ is the activation function Relu. $\mathbf{z}_{i+s,j+t}^{0}$ denotes the block of input layer for the convolution at location i, j, s, t are offsets along two axis.

Other than the sliding window mechanism, convolution also has two beneficial properties: location invariance and compositionality. Because the spatial attention layer already encapsulates all the useful positional information, extra location dependency in this phase would just add more noise to the matching result. Therefore, the aggregation method should just extract meaningful local matching signals without considering their positions, and conveniently, convolution achieves this goal. Its second property, compositionality, is coherent with the nature of languages. In English, the most basic building blocks are letters. By composing different letters together we have different words conveying different meanings. By further assembling different words together we get phrases and sentences, which could encompass more complex ideas. Convolution works in a similar fashion where small, local patches of features are merged into higher-level representations. It makes intuitive sense that CNN

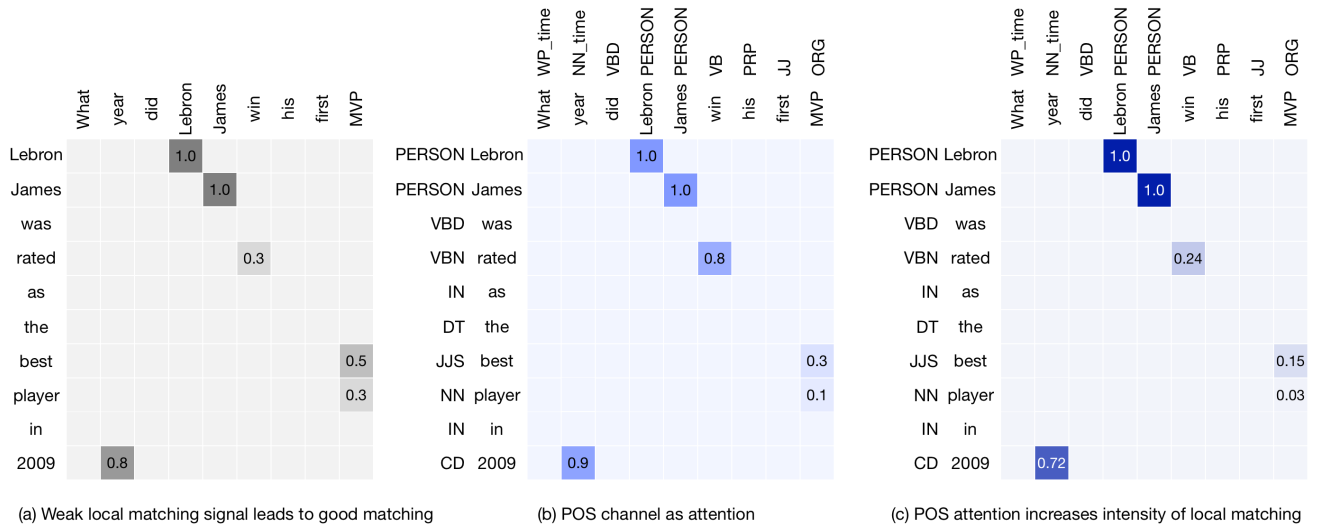


Figure 6: Case study shows weak local matching signal may also lead to good matching as the POS channel acts as an attention mechanism to better depict the relations between global matching and local matching.

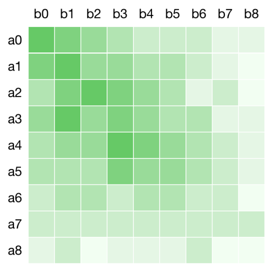


Figure 7: Element-wise intensity of spatial attention layer

hierarchically extracts features from local matching signals all the way to global matching results.

At this point, we obtain a unified, end-to-end architecture to model multi-channel matching information, and have a method of combining channels and abstract high-level features of text matching.

4 PERFORMANCE EVALUATION

We conducted performance evaluation of our new model in comparison with state-of-the-art text matching models in three scenarios: 1) offline tests based on publicly available dataset WikiQA from Microsoft [25]; 2) offline tests on Tencent's Venus computation platform with a big question-document dataset constructed from the search logs of Tencent's Mobile QQ Browser; 3) online A/B tests, with training conducted on Tencent's Venus computation platform based on big data and testing on real users.

We evaluated the following algorithms and have individually fine-tuned the performance of each for a fair comparison:

- Deep Semantic Similarity Model (DSSM) [8];
- Convolutional Deep Semantic Similarity Model (CDSSM) [18];
- Convolutional neural network architecture I of (ARC-I) [7];
- Convolutional neural network architecture II of (ARC-II) [7];
- MatchPyramid (MP) [15];
- Deep Relevance Matching Model (DRMM) [6];
- Multi-channel Information Crossing with 2×2 channels (MIX-4channel). As stated in Sec. 3.1, we compute cartesian products between unigram convolution results and bigram convolution results of two sentences in the local matching phase.
- Multi-channel Information Crossing with 3×3 channels (MIX-9channel). As stated in Sec. 3.1, we compute cartesian products between unigram convolution results, bigram convolution results, and trigram results of two sentences in the local matching phase.
- Multi-channel Information Crossing with spatial attention (MIX-spatial). Based on MIX-9channel, as stated in Sec. 3.2, we weigh matrix of local matching with the layer of spatial attention.
- Multi-channel Information Crossing with POS attention (MIX-POS). Based on MIX-spatial, as stated in Sec. 3.2, we additionally weigh matrix of local matching with the layer of POS attention. Specifically, we conduct Name Entity Recognition(NER) to further enrich the POS information. The POS and NER tags are extracted with the Natural Language Toolkit(NLTK) [1].
- Multi-channel Information Crossing with term-weight attention (MIX-weight). Based on MIX-POS, as stated in Sec. 3.2, we additionally weigh matrix of local matching with the layer of term-weight attention. Although this layer is trainable, we still initialize it with normalized products between Inverse Document Frequency(IDF) of two terms.

The algorithms are compared in terms of the number of clicks in online A/B tests, and in terms of normalized discounted cumulative gain (NDCG) and mean average precision (MAP) in offline evaluation.

NDCG@p is a popular measure of ranking quality introduced in [10]. It is also widely used in search engine performance evaluation. The idea of NDCG@p is to estimate the amount of relevant information in the first p search results, given a particular query. It also adds a penalty to the final score if highly relevant documents appear later in a search result. A NDCG@p score is always between 0 and 1, with higher values indicating better performance. In this work, we report the NDCG@3 and NDCG@5 scores.

MAP is another common metric for Information Retrieval (IR) tasks. It is build upon the Average Precision (AP) metric, where AP measures the ratio of relevant documents vs. all retrieved documents for a particular query. Now assume that we have Q queries, the MAP score is just the average of their AP scores, which is also between 0 and 1. Intuitively, a larger MAP score suggests better performance.

4.1 Offline Tests

We first present results in the offline test based on WikiQA dataset from Microsoft and query-document pair dataset from QQ Mobile Browser, respectively. WikiQA is a publicly available dataset for open-domain question answering [25]. The dataset contains 3,047 questions sampled from the query logs of Bing. Based on the clicking behavior of users, each question is associated with several answers from Wikipedia, and the total number of questions and answers are 29,258. Then, crowdsourcing workers were employed to label whether a candidate answer for a question is correct, thus 1,473 sentences are labeled as correct answers.

On the other hand, we refer to query-document pair dataset from QQ Mobile Browser as the QBSearch dataset. QBSearch dataset consists of 12,000 Chinese queries collected from query logs. Each query is associated with 10 document abstracts (within 100 words). Therefore, 120,000 pairs of query and candidate documents are sampled. Then, the documents are labeled according to whether they are correct results for a query.

For both datasets, we randomly split them into training sets and testing sets. After training the models, we generate all $\hat{y}$ from testing sets and calculate NDCG as well as MAP accordingly. The implementations of other state-of-the-art models are based on open-source project MatchZoo [4].

For the WikiQA dataset, we compare different models in terms of NDCG@3, NDCG@5 and MAP. Pairs labeled as 1 in test set are positive samples while those labeled 0 are negative samples. First, we compare our model with state-of-the-art algorithms. As shown in Table. 1, MIX model significantly outperforms other state-of-the-art algorithms in all three metrics. MIX-weight, our best performing model variant, shows improvements of 11.1% in terms of NDCG@3, 6.3% in terms of NDCG@5 and 14.6% in terms of MAP compared with MatchPyramid, which is one of the best state-of-the-arts.

Second, we evaluate the performance of MIX under different practical optimization settings and MIX shows great scalability overall. As NDCG@3, NDCG@5 and MAP are positively correlated, for convenience purposes, we use NDCG@3 as the main performance indicator. After increasing the number of local-matching layers, as stated in Sec. 3.1, MIX is able to capture more useful semantic information from each term, and the performance improved by 2.2%. After incorporating the attention layer based on spatial information, MIX learned the importance term positions in both sentences, thus the performance further improves by 2.2%. POS and NER tags are then employed to help MIX recognize important interactions with regard to grammar information. The resulting performance improves again by 3.2%. At last, we incorporate term weights in interactions into the model, by focusing more on the interactions between key words. The performance improves by another 4.2%.

Table 1: Evaluation in single-machine tests on the WikiQA dataset.

Name	NDCG@3	NDCG@5	MAP
DSSM	0.547	0.617	0.575
CDSSM	0.559	0.615	0.562
ARC-I	0.569	0.639	0.596
ARC-II	0.568	0.626	0.592
DRMM	0.619	0.670	0.622
MP	0.642	0.704	0.622
MIX-4channel	0.637	0.710	0.659
MIX-9channel	0.651	0.714	0.672
MIX-spatial	0.665	0.715	0.684
MIX-POS	0.686	0.721	0.697
MIX-weight	0.715	0.748	0.713

Table 2: Evaluation in single-machine tests on the QBSearch dataset.

Name	NDCG@3	NDCG@5	MAP
DSSM	0.617	0.706	0.650
CDSSM	0.627	0.692	0.634
ARC-I	0.654	0.732	0.671
ARC-II	0.641	0.701	0.670
DRMM	0.698	0.754	0.695
MP	0.716	0.786	0.733
MIX-4channel	0.725	0.816	0.743
MIX-9channel	0.727	0.803	0.748
MIX-spatial	0.742	0.815	0.765
MIX-POS	0.748	0.812	0.769
MIX-weight	0.775	0.820	0.792

For QBSearch dataset, we also compare different models in terms of NDCG@3, NDCG@5 and MAP. As shown in Table. 1, MIX-weight model significantly outperforms other state-of-the-art algorithms on Chinese data. However, we make the following observations from our experiments on QBSearch dataset: First, MIX-9channel fails to outperform MIX-4channel on QBSearch dataset. On this dataset, we performed segmentation beforehand as to preprocess the Chinese corpus. Unigrams were regarded as the minimal semantic units at most of the time, whereas bigrams may help in capturing phrases. However trigrams are not capable to incorporate more information, even worse it could bring in noise.

Second, Compared with MIX-spatial, incorporating POS information in MIX-POS does not bring any improvement. Similar to

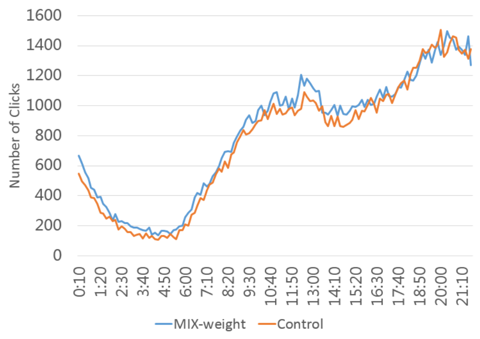


Figure 8: The number of clicks on the returned search results in online A/B testing.

searching, text matching is more about the matching of key words. Thus, matching with regard to grammar information becomes less important here. Third, As we can see, MIX-spatial shows a improvement of 2.1% compared to MIX-9channel. It makes sense as in searching tasks users tend to input key words in the beginning of their queries. The last, as mentioned above, searching task is more about the matching of key words. Therefore, by incorporating weight information, MIX-weight significantly outperforms other variants by 3.6%.

4.2 Online Tests

We ran online tests of MIX-weight in comparison to traditional IR engine, with training conducted in Venus based on query-document data collected from the search logs of the QQ Mobile Browser. We used MIX-weight as the text matching module instead of traditional IR methods. Although online performance is influenced by multiple factors, system with MIX outperforms system of control group by 5.7% in terms of the number of clicks.

When performing online split tests, the searching results generated by MIX and the original system are separately returned to two groups of randomly selected users. After 21 hours, we collected the number of clicks and conducted the following comparisons.

Fig. 8 shows MIX-model attracted a larger number of user clicks on search results within 21 hours. Since users are randomly selected, more clicks on recommended videos in a group indicate better recommendation results than other groups. Note that the improvement of MIX-model kept fading over time, this is because the original searching engine, named *Control* in the figure, can learn to rank better according to online user feedback, where good matchings are cached. Still, MIX-model demonstrated significant improvements and improved user experience in the procedure of caching.

Fig. 9 shows the reflectance and boxplot of two groups, the boxplot on the right shows that the number of clicks of MIX-weight statistically outperforms the original searching engine, which again indicates a significant improvement in performance.

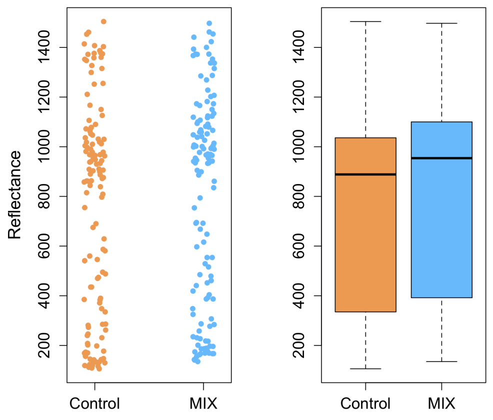


Figure 9: Reflectance(numbers of clicks per hour) of the two groups in online A/B tests and the corresponding boxplot.

5 CONCLUDING REMARKS

In this paper we propose Multi-channel Information Crossing (MIX), a novel fusion of CNNs to boost text matching quality in a real-world production environment. MIX incorporates multi-channel text features and imposes three attention layers on the key feature interactions to produce more accurate matching results. We implemented MIX on Tencent's Venus computation platform with various practical optimizations. Offline evaluations based on a publicly available dataset WikiQA and another dataset collected from real search logs of Tencent QQ Mobile Browser demonstrate that MIX outperforms many state-of-the-art methods by a substantial margin. During further online A/B split testing conducted on real users in a 21-hour period, MIX increased the Click Through Rate (CTR) by 5.7% over conventional text matching algorithms.

6 ACKNOWLEDGMENTS

The authors would like to thank Yancheng He and Gangming Jin for their guidance and support, as well as Bang Liu for many helpful discussions and insights.

REFERENCES

[1] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".
[2] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in Neural Information Processing Systems.* 737–744.
[3] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining.* ACM, 126–134.
[4] Yixing Fan, Liang Pang, JianPeng Hou, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2017. MatchZoo: A Toolkit for Deep Text Matching. *arXiv preprint arXiv:1707.07270* (2017).
[5] Matt W Gardner and SR Dorling. 1998. Artificial neural networks (the multilayer perceptron)âĂŤa review of applications in the atmospheric sciences. *Atmospheric environment* 32, 14-15 (1998), 2627–2636.
[6] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management.* ACM, 55–64.
[7] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems.* 2042–2050.
[8] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using

clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[9] Ozan Irsoy and Claire Cardie. 2014. Deep recursive neural networks for compositionality in language. In *Advances in neural information processing systems*. 2096–2104.

[10] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[11] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).

[12] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[13] Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eudard Hovy. 2015. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185* (2015).

[14] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101* (2016).

[15] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition.. In *AAAI*. 2793–2799.

[16] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering.. In *IJCAI*. 1305–1311.

[17] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[18] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 101–110.

[19] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[20] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. 129–136.

[21] Kateryna Tymoshenko, Daniele Bonadiman, and Alessandro Moschitti. 2017. Ranking Kernels for Structures and Embeddings: A Hybrid Preference and Classification Model. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 897–902.

[22] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, and Tie-Yan Liu. 2013. A theoretical analysis of NDCG ranking measures. In *Proceedings of the 26th Annual Conference on Learning Theory (COLT 2013)*.

[23] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 55–64.

[24] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 475–482.

[25] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2013–2018.

[26] Wenpeng Yin and Hinrich Schütze. 2015. Multigrancnn: An architecture for general matching of text chunks on multiple levels of granularity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 63–73.