Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 5417–5424
Marseille, 11–16 May 2020

Cross-sentence Pre-trained model for Interactive QA matching

Jinmeng Wu[a,b], **Yanbin Hao**[c]
[a]School of Electrical and Information Engineering, Wuhan Institute of Technology, China
[b]The school of Electrical Engineering, Electronics and Computer Science, The University of Liverpool, Brownlow Hill, Liverpool, UK
[c]Department of Computer Science, City University of Hong Kong, Hong Kong, 999077, China
✉ jinmeng2004910@outlook.com, haoyanbin@hotmail.com

Abstract

Semantic matching measures the dependencies between query and answer representations, which is an important criterion for evaluating whether the matching is successful. In fact, such matching does not examine each sentence individually, because the context information between sentences should be considered equally important to the syntactic context inside a sentence. Considering the above, we propose a novel QA matching model, built upon a cross-sentence context-aware architecture. Specifically, an interactive attention mechanism with a pre-trained language model is presented to automatically select salient positional answer representations that contribute more significantly to the answer relevance of a given question. In addition to the context information captured at each word position, we incorporate a quantity of context jump dependencies to leverage the attention weight formulation. This can capture the amount of useful information brought by the next word, is computed by modeling the joint probability between two adjacent word states. The proposed method is compared with multiple state-of-the-art methods using the TREC library, WikiQA, and the Yahoo! community question datasets. Experimental results show that the proposed method outperforms satisfactorily the competing ones.

Keywords: Question answering, Interactive matching, Pre-trained language model, Context jump Dependencies

1. Introduction

Question answering (QA) is the task of enabling a machine to automatically answer questions posted by humans in a natural language form. The selection of the best answer from an existing pool of candidate answers is referred to as community QA (cQA) (Shah and Pomerantz, 2010), whereas enabling the computer to automatically generate a novel answer, through some natural language models, is also known as machine dialogue (Shen et al., 2017b). In this paper, we focus on cQA by working on the semantic matching between question and answer texts. In general, semantic matching requires the accurate modeling of the relevance between two portions of text, and, in addition to QA, is widely used for tasks, such as paraphrase identification (Cheng and Kartsaklis, 2015), machine translation (Bahdanau et al., 2015), image caption generation (Karpathy and Fei-Fei, 2015) and video hyperlinking (Hao et al., 2019b).

In order to compute an accurate measure of relevance between the question and answer, it is beneficial to take the lexical, syntactic and semantic information of the text pairs into account. Traditional matching seeks effective ways of extracting semantic features to improve a given similarity metric (Meng et al., 2013). Recent advances have managed to replace this manual feature engineering process with a model that automatically learns distributed representations of words and sentences via neural networks (Wan et al., 2016; Hao et al., 2019a).

As previously mentioned, the goal of a cQA matching task is to select the correct answers from a set of candidate answers based on the content of a given question. One of the key hindrances in this, is that the key lexical components and information might not be shared between question and answer texts. In some cases, ambiguous contents in questions or answers may impede this process. Hence, the question-answer matching process can become more effective when the sentence representations for questions and answers are learned interactively, rather than in isolation.

Question Q_1

- What is the color of that cat?

Candidate Answers: A_{31} and A_{32}

- A_{11}: The cat was sitting on a mat.
- A_{12}: The cat that was sitting on a mat was red.

Answer	Key Components
A_{11}	The ***cat*** was sitting on a mat.
A_{12}	The ***cat*** that was sitting on a mat was ***red***.

Figure 1: cQA Example scenario 1 with 'cat' case.

Fig.1 shows a cQA scenario, where the aim is to answer the question, each with their own pool of answers. We highlight the key components for the answer in the figure. In both examples, these salient components in the answers directly reflect or respond to the context of the questions, which contribute more significantly towards the relevance of the given question. Such salient information or the key components in sentences can be captured by an attractive approach called attention mechanism (Bahdanau et al., 2015). This mechanism has been mostly used in the tasks related to translations (Bahdanau et al., 2015). Recent findings, for example in (Tan et al., 2016; Hermann et al., 2015), have demonstrated their applicability in assigning degrees of importance, known as attention weights, to different word positions in a sentence. In the cQA community, it has been of increasing interest to develop effective ways of building attention mechanisms, so that the matching per-

formance can be improved by learning from and paying more attention to salient text (Tan et al., 2016; Xiong et al., 2017).

In this paper, we address the aspects that have been highlighted above, and propose a novel approach to improve the standard of the response accuracy. In particular, we make the following key contributions:

1. We extend the notion of interactive learning by developing a cross-sentence context-aware bi-directional LSTM model, where we generate the hidden representations for both the question and answer texts, thereby making them aware of each other's context. As such, in the proposed model, the hidden representation for the question text, and particularly the state values for each word position, is affected not only by its previous or next states, but also by the multi-positional representations of the answer text.

2. As the interaction between question and answer texts is bi-directional, the content of the question text should also affect the way that the answer text is encoded or characterized. We propose an interactive attention mechanism for answer representation learning, and augment our proposed approach to consider the relationship between adjacent words, instead of simply concatenating the word representations. A new quantity, referred to as the context jump dependencies, is proposed to represent the joint probability between adjacent words.

3. We perform an exhaustive evaluation for the proposed approach using three community datasets, namely TREC, Yahoo! and WikiQA, and share our findings.

The remaining of the paper is organized as follows: In the Section 2., we present the proposed method, explaining the structure of proposed model. This is followed by a detailed discussion on the experimental results in Section 3.. We finally conclude the paper in Section 4..

2. Proposed Method

One common matching strategy is to first learn representations for question and answer sentences separately based on their content, and then compute a similarity score using the learned representations. Given, however, the fact that sentence matching does not examine each sentence individually, context information between different sentences should also be considered equally important to the syntactic level of the context within a single sentence. This motivates us to design a matching model built upon a cross-sentence context-aware bi-directional LSTM architecture with interactive pre-trained attention mechanism, referred to as IAM. The proposed model learns the sentence hidden representation at each state (word position) not only based on its previous (or next) state, but also the multi-positional representations of an other sentence. In this way, a context-aware sentence representation that is self-adaptive to the corresponding content is learned. Since the interaction between question and answer is mutual, we also introduce an adaptive answer representation by taking into account the question content to enhance the matching. In the following subsections, we describe in detail the proposed model; its overall architecture is illustrated in Fig.2.

2.1. Context-Aware Matching

We denote a sentence as $\boldsymbol{x} = \{x_1, x_2, \ldots, x_T\}$ where x_t is the t-th word in the sentence. A language model that uses RNN to learn a sentence representation $\mathbf{h}_t \in \mathbb{R}^K$, and it contains word context information accumulated up to the t-th word, computed from the current word representation $\boldsymbol{w}_t$. In this work, the Bi-LSTM architecture is used for generating sentence representation, where the input is a sequence of words in the sentence. We denote the hidden representation of the t-th word position in the forward direction of Bi-LSTM by the column vector $\overrightarrow{\boldsymbol{h}}_t$ while the vector $\overleftarrow{\boldsymbol{h}}_t$ for the reverse direction. We use the symbols q and a to distinguish a question sentence and an answer sentence respectively, for example, $\mathbf{h}_t^{(q)} = \left\{ \overrightarrow{\boldsymbol{h}}_t^{(q)}; \overleftarrow{\boldsymbol{h}}_t^{(q)} \right\}$ for question representation, and $\mathbf{h}_t^{(a)} = \left\{ \overrightarrow{\boldsymbol{h}}_t^{(a)}; \overleftarrow{\boldsymbol{h}}_t^{(a)} \right\}$ for answer representation, where $t = 1, 2, \ldots, T$.

We use the contextual representations to encode the final distributed matching degree vector $\mathbf{s}$ between the question and answer sentences

$$\mathbf{s} = \tanh\left(\mathbf{U}_q \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_T^{(q)} \\ \overleftarrow{\boldsymbol{h}}_T^{(q)} \end{bmatrix} + \mathbf{U}_a \sum_{t=1}^{T} \alpha_t^{(qa)} \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_t^{(a)} \\ \overleftarrow{\boldsymbol{h}}_t^{(a)} \end{bmatrix} + \mathbf{b}_s \right). \tag{1}$$

The weight matrices $\mathbf{U}_q, \mathbf{U}_a \in \mathbb{R}^{H \times 2K}$ (K is the dimensionality of sentence representation) and bias vector $\mathbf{b}_s \in \mathbb{R}^H$ are the model variables to be learned. Here, the answer representation is learned by considering query content. Particularly, the resulted embedding $\boldsymbol{h}_T^{(q)}$ of Bi-LSTM at last time T is for the question representation. While, for the answer representation, we fuse the multiple positional representations of words by a weighted sum operation, where each weight $\alpha_t^{(qa)}$ is used to control the importance degree of the combined representation contributing to its relevance to a given question. Working with the distributed similarity vector $\boldsymbol{s}$ as obtained by Eq. (1), the sentence matching task can be formulated as a binary classification problem. The probability that an answer is related to a question can be modeled using a two-way softmax function based on the computed similarity vector $\mathbf{s}$, given as

$$p(y = 1|\mathbf{s}) = \frac{\exp\left(\mathbf{s}^T \boldsymbol{\vartheta}_1\right)}{\exp\left(\mathbf{s}^T \boldsymbol{\vartheta}_0\right) + \exp\left(\mathbf{s}^T \boldsymbol{\vartheta}_1\right)}, \tag{2}$$

where the two column vectors $\boldsymbol{\vartheta}_0$ and $\boldsymbol{\vartheta}_1$ are softmax parameters with the same dimensionality as $\mathbf{s}$. Based on the above formulation, model variables can be optimized by minimizing a regularized cross-entropy cost by following the logistic regression model (Bengio, 2009; Dreiseitl and Ohno-Machado, 2002). We detail all the components below, including the learning of standard Bi-LSTM, the question representation learning, answer representation learning and the computation of attention weight $\alpha_t^{(qa)}$.

2.2. Standard Bi-LSTM Learning

The column vectors $\mathbf{h}_t^{(q)}$ and $\mathbf{h}_t^{(a)}$ are initially computed from two Bi-LSTMs, storing a hidden representation en-

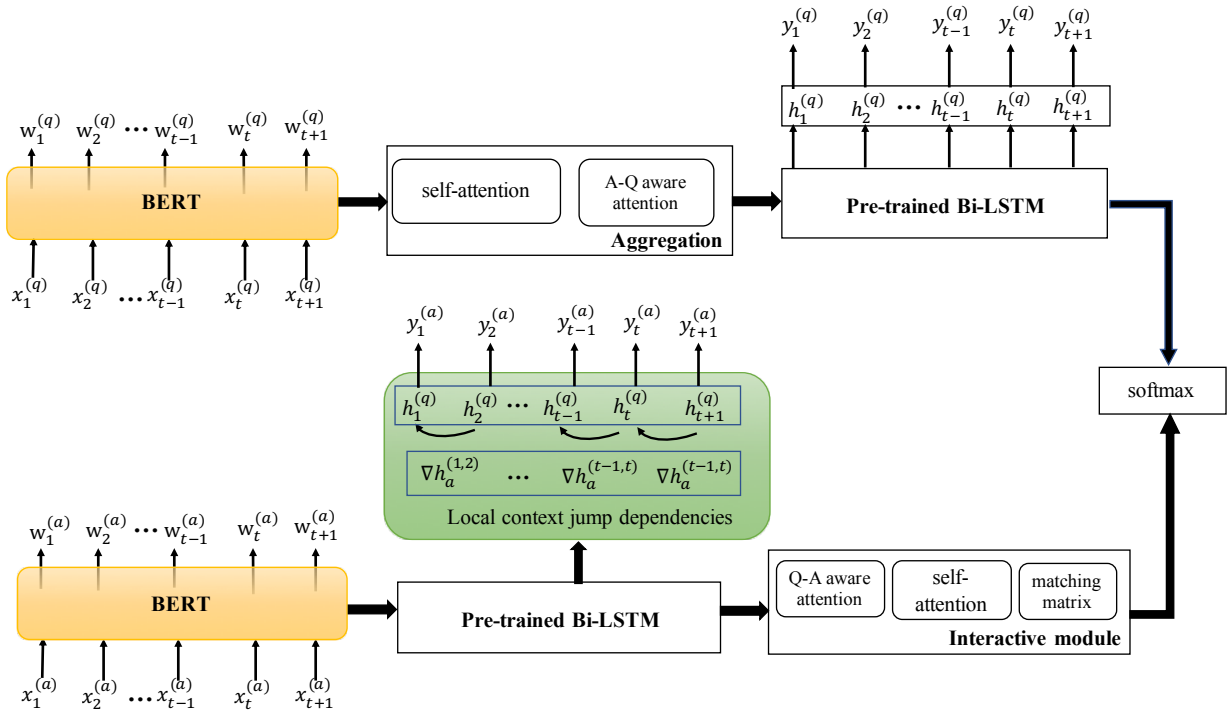

Figure 2: Illustration of the proposed IAM model architecture.

coding the question/answer content. These vectors are obtained by training a standard Bi-LSTM over a sentence generation task, assisted by the BERT model (Devlin et al., 2018). Here, we pre-train the parameters of Bi-LSTM by using the probabilistic language model (Bengio et al., 2003), the probability of generating a question sentence $\boldsymbol{x}_q = \left\{x_1^{(q)}, x_2^{(q)}, \ldots, x_T^{(q)}\right\}$ is modeled by

$$p(\boldsymbol{x}_q) = \prod_{t=1}^{T} \frac{\exp\left(\mathbf{W}_t^{(q)} \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_t^{(q)} \\ \overleftarrow{\boldsymbol{h}}_t^{(q)} \end{bmatrix} + \mathbf{b}_t^{(q)}\right)}{\sum_{i=1}^{T} \exp\left(\mathbf{W}_i^{(q)} \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_i^{(q)} \\ \overleftarrow{\boldsymbol{h}}_i^{(q)} \end{bmatrix} + \mathbf{b}_i^{(q)}\right)}. \quad (3)$$

The weight matrices $\left\{\mathbf{W}_t^{(q)}\right\}_{t=1}^{T}$ and bias terms $\left\{\mathbf{b}_t^{(q)}\right\}_{t=1}^{T}$ are model variables to be optimized. Working with the above probability function, model optimization relies on maximizing the log-likelihood over a training corpus of question sentences. Moreover, $\mathbf{h}_t^{(q)}$ further considers its interaction with the answer, where the answer hidden representation $\mathbf{h}_t^{(a)}$ is used in the attention module.

2.3. Question-aware Representation Learning

To compute a context-aware question sentence representation that is self-adaptive to the answer content, we modify the Bi-LSTM activation function, so that the current state $\overrightarrow{\boldsymbol{h}}_t^{(q)}$ is computed not only from the previous state of the current word, but also the answer content. This corresponds to the following set of equations

$$\begin{bmatrix} \overrightarrow{\boldsymbol{h}}_t^{(q)} \\ \overleftarrow{\boldsymbol{h}}_t^{(q)} \end{bmatrix} = \text{Bi-LSTM}\left(\boldsymbol{\phi}_t^{(qa)}, \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_{t-1}^{(q)} \\ \overleftarrow{\boldsymbol{h}}_{t-1}^{(q)} \end{bmatrix}\right), \quad (4)$$

where

$$\boldsymbol{\phi}_t^{(qa)} = \tanh\left(\mathbf{V}_q \mathbf{w}_t^{(q)} + \mathbf{V}_a \sum_{t=1}^{T} \boldsymbol{\alpha}_t^{(qa)} \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_t^{(a)} \\ \overleftarrow{\boldsymbol{h}}_t^{(a)} \end{bmatrix} + \mathbf{b}_\phi\right). \quad (5)$$

The weight matrices $\mathbf{V}_q$ and $\mathbf{V}_a$ and the bias vector $\mathbf{b}_\phi$ are the variables to be optimized. The question word vector $\mathbf{w}_t^{(q)}$ is pre-trained by the BERT. Through adding the answer content, the input of the above model is changed from the current word vector $\boldsymbol{w}_t^{(q)}$ to a richer vector $\boldsymbol{\phi}_t^{(qa)}$. For notational convenience, let

$$\boldsymbol{c}_a = \sum_{t=1}^{T} \boldsymbol{\alpha}_t^{(qa)} \begin{bmatrix} \overrightarrow{\boldsymbol{h}}_t^{(a)} \\ \overleftarrow{\boldsymbol{h}}_t^{(a)} \end{bmatrix}. \quad (6)$$

This vector $\boldsymbol{c}_a$ is designed to be adaptive to the question content controlled by its weights $\left\{\boldsymbol{\alpha}_t^{(qa)}\right\}_{t=1}^{T}$. It has the same function in Eqs. (1) and (5). The answer representations $\overrightarrow{\boldsymbol{h}}_t^{(a)}$ and $\overleftarrow{\boldsymbol{h}}_t^{(a)}$ are computed from another Bi-LSTM. Now, the remaining question is how to construct the weights $\boldsymbol{\alpha}_t^{(qa)}$ to allow the question content to affect the importance of each positional answer representation. For this we propose an interactive attention mechanism is explained in Section 2.4. firstly.

2.4. Pre-trained Interactive Attention

The answer context vector $\boldsymbol{c}_a$ in Eq. (6) is proposed to fuse within-sentence and cross-sentence context, aiming at serving better the task of question answer matching. The syntactic level of context within the answer sentence is encoded by its positional representations, which combine $\overrightarrow{\boldsymbol{h}}_t^{(a)}$ and $\overleftarrow{\boldsymbol{h}}_t^{(a)}$ computed by the standard Bi-LSTM, for $t = 1, 2, \ldots, T$. To automatically discover the keyword

positions that capture better the answer relevance to a given question, we construct the weight function as follows

$$\alpha_t^{(qa)} = \frac{\exp\left(\boldsymbol{e}_t^{(qa)}\right)}{\sum_{i=1}^{T}\exp\left(\boldsymbol{e}_i^{(qa)}\right)}, \tag{7}$$

where the energy function $\boldsymbol{e}_t^{(qa)}$ is formulated as

$$\boldsymbol{e}_t^{(qa)} = \tanh\left(\boldsymbol{u}_h^T\begin{bmatrix}\overrightarrow{\boldsymbol{h}}_t^{(q)}\\ \overleftarrow{\boldsymbol{h}}_t^{(q)}\end{bmatrix} + \boldsymbol{u}_h^T\begin{bmatrix}\overrightarrow{\boldsymbol{h}}_t^{(a)}\\ \overleftarrow{\boldsymbol{h}}_t^{(a)}\end{bmatrix} + v_1\nabla h_a^{(t,t+1)} + \boldsymbol{u}_a^T\mathbf{d}_a + v_2\mathbf{d}_q^T\mathbf{M}\mathbf{d}_a\right). \tag{8}$$

The column $\boldsymbol{u}$ vectors with different subscript symbols, the matrix $\mathbf{M}$ and the scalars $v_{1,2}$ are model variables to be optimized. These weight functions $\left\{\boldsymbol{e}_t^{(qa)}\right\}_{t=1}^{T}$ select salient positional representations more relevant to the question context, and are referred to as attention weights (Bahdanau et al., 2014). As seen in Eq. (8), there are different types of information that contribute to the computation of $\boldsymbol{e}_t^{(qa)}$. The answer and question content jointly controls the selection of the salient representation, and comprises the interactive attention mechanism. Naturally, the characteristic of each positional representation itself $\begin{bmatrix}\overrightarrow{\boldsymbol{h}}_t^{(a)}\\ \overleftarrow{\boldsymbol{h}}_t^{(a)}\end{bmatrix}$ contributes to its own importance degree. Additionally, we consider other quantities, such as $\nabla h_a^{(t,t+1)}$, $\mathbf{d}_q$ and $\mathbf{d}_a$. Specifically, the column vectors $\mathbf{d}_q, \mathbf{d}_a \in \mathbb{R}^D$ are bag-of-word representations of the question and answer sentences, with D denoting the size of the word vocabulary. Each dimension of the vector corresponds to the term frequency - inverse document frequency (tf-idf) of the corresponding word. The bilinear score $\mathbf{d}_q^T\mathbf{M}\mathbf{d}_a$ examines relevance between sentences based on their shared words weighted by a set of between-word interaction scores that are stored as elements of the word similarity matrix $\mathbf{M}$. The incorporation of basic word frequency information (via $\mathbf{d}_q$ and $\mathbf{d}_a$) and word co-occurrence information (via $\mathbf{d}_q^T\mathbf{M}\mathbf{d}_a$) is motivated by the loss of specific information (such as years and proper nouns), which may not be accounted for in the distributed representation of the words, but is useful when matching a query to an answer.

2.5. Local Context Jump Dependencies

In this section, we introduce the quantity $\nabla h_a^{(t,t+1)}$ that participates in the attention weight computation of Eq. (8). When using a Bi-LSTM to learn the sentence representation, each obtained positional representation accumulates context information up to the targeted word position in a sentence. It is reasonable to assume that if the subsequent word brings significant change to the sentence semantics, it can directly affect the importance degree of the positional representation at the current word. Such a change in sentence semantics could be indicated by the information change contained within the learned hidden representations between the current and next states.

Therefore, given an answer sentence, we aim to formulate a quantity $\nabla h_a^{(t,t+1)}$ that can be potentially used as an indicator of its information change between the current (t) and the next ($t+1$) word positions. It is known that the joint entropy between the positional representations of two adjacent states measures the uncertainty associated with the two word positions. The amplitude of such uncertainty is a good indicator of the amount of new information to be brought by the next word. Thus, we attempt to estimate the joint probability of the answer representations at the current and next word positions, which fundamentally decides the joint entropy between the two states.

Given a set of learned positional representations for an answer sentence, denoted by $\mathbf{h}_t^{(a)}$ for $t = 1, 2, \ldots, T$. The joint probability of the two adjacent states in a sentence is estimated by following a similar model architecture to RBM (LeCun et al., 2006; Salakhutdinov and Hinton, 2009). By treating the two adjacent states as the observable units, their joint probability can be modeled as

$$p\left(\mathbf{h}_t^{(a)}, \mathbf{h}_{t+1}^{(a)}|\boldsymbol{\theta}\right) = \frac{1}{Z(\boldsymbol{\theta})}\exp\left(-E\left(\mathbf{h}_t^{(a)}, \mathbf{h}_{t+1}^{(a)}, \boldsymbol{\theta}\right)\right), \tag{9}$$

with

$$E\left(\mathbf{h}_t^{(a)}, \mathbf{h}_{t+1}^{(a)}, \boldsymbol{\theta}\right) = \left(\mathbf{h}_t^{(a)}\right)^T\mathbf{Q}\mathbf{h}_{t+1}^{(a)} + \mathbf{q}_1^T\mathbf{h}_t^{(a)} + \mathbf{q}_2^T\mathbf{h}_{t+1}^{(a)}, \tag{10}$$

where $\boldsymbol{\theta} = \{\mathbf{Q}, \boldsymbol{q}_1, \boldsymbol{q}_2\}$ is the set of hidden units to be estimated. The partition function $Z(\boldsymbol{\theta})$ is used to normalize the joint probabilities, so that they sum to 1 over all the state possibilities. Let $\mathbf{h}_t^{(a_i)}$ and $\mathbf{h}_{t+1}^{(a_i)}$ denote the t-th and $(t+1)$-th states of the i-th answer sentence. The log-likelihood $L(\boldsymbol{\theta})$ of a set of observed example pairs of adjacent states, constructed from a training corpus of answer sentences $\{a_i\}_{i=1}^{M}$, is maximized, given as

$$L(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}\sum_{t=1}^{T-1}\log\left(p\left(\mathbf{h}_t^{(a_i)}, \mathbf{h}_{t+1}^{(a_i)}|\boldsymbol{\theta}\right)\right). \tag{11}$$

In the above, M denotes the total number of answer sentences used for training, and maximization of the log-likelihood involves computing the partition function $Z(\boldsymbol{\theta})$ and its partial derivative. This is difficult due to the fact that samples cannot be drawn directly from $p\left(\mathbf{h}_t^{(a_i)}, \mathbf{h}_{t+1}^{(a_i)}|\boldsymbol{\theta}\right)$ as the value of the partition function is unknown. Instead, we employ the contrastive divergence method (Hinton, 2002), and estimate the variable change in each update according to

$$\triangle\boldsymbol{\theta}_{k+1} \propto \frac{\partial E\left(\mathbf{h}_t^{(a)}, \mathbf{h}_{t+1}^{(a)}, \boldsymbol{\theta}_k\right)}{\partial\boldsymbol{\theta}} - \frac{E\left(\widehat{\mathbf{h}_t^{(a)}}, \widehat{\mathbf{h}_{t+1}^{(a)}}, \boldsymbol{\theta}_k\right)}{\partial\boldsymbol{\theta}}, \tag{12}$$

where $\boldsymbol{\theta}_k$ denotes the estimated model variables at the k-th iteration, and $\triangle\boldsymbol{\theta}_{k+1}$ denotes the approximated variable change for the update in the next iteration. The reconstructed representations $\widehat{\mathbf{h}_t^{(a)}}$ and $\widehat{\mathbf{h}_{t+1}^{(a)}}$ are obtained by following multiple cycles of Markov chain Monte Carlo (MCMC) sampling (Andrieu et al., 2003) based on the current variable estimate $\boldsymbol{\theta}_k$. Given the final estimated model

variables $\boldsymbol{\theta}^*$, we define the quantity $\nabla h_a^{(t,t+1)}$ as

$$\nabla h_a^{(t,t+1)} = \exp\left(-E\left(\mathbf{h}_t^{(a)}, \mathbf{h}_{t+1}^{(a)}, \boldsymbol{\theta}^*\right)\right). \quad (13)$$

Because this quantity is used as an indicator of the degree by which new information is conveyed by the next word between two adjacent states of an answer sentence, we refer to it as context information jump.

2.6. Model Training and Initialization

In this section, we briefly provide some implementation details for the model training and initialization phases. Given a training corpus for question answer matching, we first train the Bi-LSTM model using the question sentences and answer sentences, independently. The model is trained to solve the language generation task via log-likelihood maximization based on the sentence generation probabilities, as formulated by Eq. (3). Sentence representations are learned by the language model, such as $\left\{\boldsymbol{h}_t^{(q)}\right\}_{t=1}^{T}$ and $\left\{\boldsymbol{h}_t^{(a)}\right\}_{t=1}^{T}$, acting as fixed input of the proposed matching model. By minimizing a regularized cross-entropy cost, the matching model is trained to solve a binary classification problem that decides the relevance. Instead of random initialization, we initialize all the distributed word representation vectors ($\mathbf{w}_t^{(q)}$ and $\mathbf{w}_t^{(a)}$) by the word vectors trained by the BERT language model (Devlin et al., 2018). Cosine similarities computed from these word vectors are also used to initialize the word similarity matrix $\mathbf{M}$. The LSTM variables used to compute the sentence representation is initialized by the previously trained Bi-LSTM model. The remaining variables in the matching model are initialized randomly.

3. Experimental Results and Analysis

We have proposed a cross-sentence context-aware Bi-LSTM architecture with interactive pre-trained attention mechanism, referred to as IAM. In this section, it is compared and evaluated with various state-of-the-art neural matching models using three benchmark cQA datasets. In addition to the performance comparisons in Table 1, we provide various examples in Tables 2-4 to offer insights on the intermediate results learned by different models.

3.1. Datasets and Experimental Setup

The dataset TREC is generated from TREC QA tracks 8-13, containing a set of factoid queries and candidate answers (Wang et al., 2007). The correct answers for each query are manually labeled and ranked in the dataset. Three partitions of TRAIN, DEV and TEST are provided in the data. Following the benchmark evaluation scheme as used in existing work, two experiments are conducted in which the TRAIN is used to train separate models. In both experiments, DEV is used for model validation and TEST to report the answer selection performance.

The Yahoo! answer collection is a large-scale dataset collected through Yahoo! Webscope Program[1] based on community service. It includes approximately 4 million questions and answers, and each question is associated with a best answer and a category. The BM25 retrieval algorithm[2] is used to retrieve the top 100 answers for each question. These retrieved answers are also labeled as the correct ones for each corresponding question, ranked after its best answer provided by the collection (Zhou et al., 2016).

WikiQA is a new released question answering dataset on open-domain area. All questions in dataset are sampled from query logs of Bing website, which are posted by users. Each question is related to a Wikipedia website that potentially has correct answers. The candidate answers are collected from Wikipedia website (Yang et al., 2015).

To process the datasets, a special end-of-sentence symbol $\langle$EOS$\rangle$ is added to the end of each sentence, and the out-of-vocabulary words are mapped to a special token symbol $\langle$UNK$\rangle$. We follow the same text pre-processing procedure as in (Severyn and Moschitti, 2015). The used Bi-LSTM architecture contains two layers each with 100 hidden units. Dimensionality of each word embedding vector is set to $K = 100$. To initialize the word embedding vectors for the TREC data, a wor2vec model (Mikolov et al., 2013) is trained using the Wikipedia dumps[3] containing approximately 3 million words (after removing words that appear less than 5 times in the corpus). The learning rate is set to 0.025. To initialize the word embedding vectors for the Yahoo! cQA dataset, the Glove model[4] is trained using the 2B Tweets corpus containing approximately 1.2 million words after removing the infrequent ones (Pennington et al., 2014). For words do not appear in Wikipedia (or Tweeter), a random value uniformly sampled from the interval $[-0.3, 0.3]$ is assigned to each embedding dimension. For other model variables to be initialized, random values uniformly sampled from the interval $[-0.05, 0.05]$ are used. Stochastic gradient descent is used for model optimization with a mini-batch containing 50 training examples, a learning rate of 0.1, and a dropout rate of 0.5 (Srivastava et al., 2014). The learning rate is decreased by a factor of 0.5 after 10 epochs. Gradient clipping (Pascanu et al., 2013) is used to scale the gradient when the norm of gradient exceeds a threshold of 5. The performance of existing methods is reported using the implementation settings provided in their corresponding published papers, and with the same training, validation and test data split as the proposed method.

3.2. Results and Analysis

To report model performance using the test set, we use three performance metrics, namely mean reciprocal rank (MRR), mean average precision (MAP). We collected the reported results from the published works of the compared models, wherever possible. Wherever this was not feasible, we implemented them to match with the reported specification and experimental evaluation. Performance comparison between the proposed IAM and multiple compared neural matching models are reported in Table 1 for the TREC, Wiki and Yahoo! datasets. It can be seen that IAM achieves the best performance for both datasets in terms of multiple performance measures. There is a significant performance

[1] http://webscope.sandbox.yahoo.com

[2] https://lucene.apache.org/

[3] http://dumps.wikimedia.org/backup-index.html

[4] http://nlp.stanford.edu/projects/glove/

Table 1: Performance comparison of different models across a range of datasets. The best results are highlighted and the second best results are underlined.

	TREC		WikiQA		Yahoo!	
Models	MRR	MAP	MRR	MAP	MRR	MAP
QA-LSTM (Tan et al., 2016)	0.8322	0.7111	0.7045	0.6821	0.6468	0.6157
AP-CNN (Santos et al., 2016)	0.8511	0.7530	0.6957	0.6886	0.6489	0.6047
Ab-CNN (Yin et al., 2016)	0.8539	0.7741	0.7108	0.6921	0.6530	0.6325
KV-MemNNs (Miller et al., 2016)	0.8523	0.7857	0.7265	0.7069	0.6749	0.6431
IARNN (Wang et al., 2016)	0.8208	0.7369	0.7418	0.7341	0.6687	0.6275
BiMPM (Wang et al., 2017)	0.8750	0.8020	0.7310	0.718	0.6892	0.6353
IWAN (Shen et al., 2017a)	0.8890	0.8220	0.7500	0.7330	0.7010	0.6521
CAM (Wang and Jiang, 2017)	0.8659	0.8145	0.7545	0.7433	0.7035	0.6630
ELMo (Peters et al., 2018)	0.8810	0.8247	0.7430	0.7369	0.7163	0.6758
BERT-based (Devlin et al., 2018)	0.8827	0.8263	0.7592	0.7457	0.7218	0.6792
IAM+BERT	**0.8951**	**0.8426**	**0.7631**	**0.7520**	**0.7362**	**0.6831**

Table 2: Comparison of the top three answers returned by the proposed and the existing architecture for the first example question.

Example 1	
Question	What can I do with fresh banana peppers and habanero peppers?
Top 3 answers by proposed	**No.1**: Dry them and use then through out the year. You can also use them raw or cooked, but I would dry them and then when you need one or two then just rehydrate in water.
	No.2: As for the habaneros, be careful when handling them, these are some of the hottest peppers known and they make great jellies.
	No.3: I've been growing peppers in my garden this year. I've just got a good crop of them.
Top 3 answers by BERT-based (Devlin et al., 2018)	**No.1**: I've been growing peppers in my garden this year. I've just got a good crop of them.
	No.2: Dinner that will go good with pasta salad. Any of your favourite meats: ribs, chicken, pork chops.
	No.3: As for the habaneros, be careful when handling them, these are some of the hottest peppers known and they make great jellies.

Table 3: Comparison of the top three salient word positions in answer captured by the proposed method and the attention Bi-LSTM . The salient words are highlighted in bold with their corresponding attention weights indicated in parenthesis next to the words.

Example 1	
Question	How do I know when steamed salmon is done?
Proposed	I steamed the ***salmon*** (0.367) as directed. A fillet maybe under a pound for ***18 mins*** (0.431) and it has white liquid ***spots*** (0.586) on top of the fish.
Attention Bi-LSTM (Tan et al., 2016)	I steamed the ***salmon*** (0.373) as directed. A fillet maybe ***under*** (0.302) a pound for 18 mins and it has white liquid spots on top of the ***fish*** (0.462).
Example 2	
Question	How do glass touchscreens work?
Proposed	I'm not sure but I think there's a huge ***chip*** (0.578) beneath the layer of ***glass*** (0.513) that sends waves of information about where you touched. The glass, being an electric ***insulator*** (0.475), repels it toward the huge chip inside.
Attention Bi-LSTM (Tan et al., 2016)	I'm not sure but I think there's a huge chip beneath the layer of ***glass*** (0.385) that sends waves of ***information*** (0.436) about where you touched. The glass, being an electric insulator, repels it toward the ***huge*** (0.329) chip inside.

improvement of more than 7% over the CNN-based matching models, and an improvement of around 1-2% improvement over the state-of-the-art BERT-based transformer architecture. In Table 2, we compare the top three answers of the example question returned by the proposed model and the BERT-based transformer model (Devlin et al., 2018). The two models return two same sentences in their top three list in the first example of Table 2. For the example, the rankings produced by IAM are more accurate.

Compared to the existing attention mechanism used by the attention Bi-LSTM baseline model (Tan et al., 2016), the proposed one provides better matching performance. In Table 3, we illustrate the learned salient word positions of two example answer sentences, which are interactively controlled by both question and answer content. Attention weights learned by the proposed and existing attention mechanisms are compared for the same pair of question and answer. It can be seen from Table 3, that the proposed

Table 4: Illustration of extreme word positions in the answer with either the largest two values of $\nabla h_a^{(t,t+1)}$ indicated by T@K for K=1,2 (highlighted by bold), or the smallest two values of $\nabla h_a^{(t,t+1)}$ indicated by B@K for K=1,2 (underlined). We use $\mathbf{Q}$, $\mathbf{A}_+$ and $\mathbf{A}_-$ to distinguish the question, correct answer and incorrect answer sentences.

Example 1	
Q: How do I remove tag the tag names on Facebook?	
$\mathbf{A}_+$: ***View*** (T@2) video and click the "***remove*** (T@1) tag" link <u>*next*</u> (B@1) to that person's name. It will no longer be <u>*linked*</u> (B@2) to their profile.	$\mathbf{A}_-$: It ***allows*** (T@2) scientists to ***trace*** (T@1) evolution of <u>*species*</u> (B@2) based on the mutations of their genetic code, as well as looking for <u>*new*</u> (B@1) ones.
Example 2	
Q: What percentage of alcohol freezes?	
$\mathbf{A}_+$: Vodka is ***num*** (T@2) ethanol (and num water). <u>*When*</u> (B@1) a substance is dissolved in water, it lowers the ***freezing*** (T@1) point <u>*of*</u> (B@2) the solution.	$\mathbf{A}_-$: Well I was watching a ***cooking*** (T@1) show, (jamies family christmas) and he cut the top <u>*off*</u> (B@2) of a ***plastic*** (T@1) bottle and put a bottle of vodka inside it, he then filled the remaining space with water, then he put it <u>*in*</u> (B@1) the freezer.

method is able to capture more accurately the salient word positions that are important for the matching task.

To examine the importance of incorporating the quantity of context information jump $\nabla h_a^{(t,t+1)}$ to the attention calculation, the proposed model is trained on TRAIN-ALL set with $\nabla h_a^{(t,t+1)}$ removed for the TREC data. This leads to a performance drop of 1.2%-1.55%, compared to the complete IAM as shown in Table 1. This indicates that it is effective to take into account information change at the targeted word position when evaluating its contribution to the whole sentence semantics. To illustrate the effect of this quantity, we highlight some word positions possessing either very high or very low values of $\nabla h_a^{(t,t+1)}$ in one correct and one incorrect answer of a given question. Two such examples are displayed in Table 4. It is interesting to observe that some highlighted word positions indeed contribute significantly to the question/answer relevance.

4. Conclusion

We have proposed the question answer sentence matching model IAM, based on a cross-sentence context-aware bi-directional LSTM architecture with interactive attention mechanism. It improves semantic matching between sentences by taking into account context information, in addition to the syntactic level of context within the sentence. IAM learns the hidden representation at each word position for a question, not only based on the previous (or next) state, but also the multi-positional representations of the answer sentence. This results in a context-aware question representation, self-adaptive to the answer content. A novel attention mechanism is designed to generate an answer representation adaptive to the question content. In addition to the positional answer and question representations as used in most existing attention mechanisms, we include word frequency and co-occurrence information to the attention weight computation. More importantly, we propose the new quantity of context information jump to improve the attention computation by taking into account information changes at different word positions. The proposed model is compared to various neural matching models, based on CNN or RNN architectures. IAM outperforms all the competing ones for both TREC and Yahoo cQA datasets, as evidenced by multiple matching performance measures. We also provide illustrating examples, such as the top selected answers and automatically learned salient word positions, to demonstrate the effectiveness of the proposed method.

5. Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 61671337).

6. Bibliographical References

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proceedings the 6th ICLR International Conference on Learning Representations*.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. 3:1137–1155.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127.

Cheng, J. and Kartsaklis, D. (2015). Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proceedings of the 2015 EMNLP Conference on Empirical Methods in Natural Language Processing*, pages 1531–1542.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Dreiseitl, S. and Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, 35(5):352–359.

Hao, Y., Mu, T., Hong, R., Wang, M., Liu, X., and Goulermas, J. Y. (2019a). Cross-domain sentiment encoding through stochastic word embedding. *IEEE Transactions on Knowledge and Data Engineering*.

Hao, Y., Ngo, C.-W., and Huet, B. (2019b). Neighbourhood structure preserving cross-modal embedding for

video hyperlinking. *IEEE Transactions on Multimedia*, 22(1):188–200.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Proceedings of the 28th NIPS Conference on Advances in Neural Information Processing Systems*, pages 1693–1701.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. (2006). A tutorial on energy-based learning. *Predicting Structured Data*, 1.

Meng, L., Huang, R., and Gu, J. (2013). A review of semantic similarity measures in wordnet. *International Journal of Hybrid Information Technology*, 6(1):1–12.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv: 1301.3781*.

Miller, A., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A., and Weston, J. (2016). Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *Proceedings of the 30th ICML International Conference on Machine Learning*, volume 28, pages 1310–1318.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of ACL-EMNLP Conference on Empirical Methods in Natural Language Processing*, volume 14, pages 1532–43.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Salakhutdinov, R. and Hinton, G. E. (2009). Deep boltzmann machines. In *Proceedings of the 21st AISTATS International Conference on Artificial Intelligence and Statistics*, volume 1, page 3.

Santos, C. d., Tan, M., Xiang, B., and Zhou, B. (2016). Attentive pooling networks. In *CoRR*, volume abs/1602.03609.

Severyn, A. and Moschitti, A. (2015). Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382, New York, NY, USA.

Shah, C. and Pomerantz, J. (2010). Evaluating and predicting answer quality in community QA. In *Proceedings of the 33rd ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 411–418.

Shen, G., Yang, Y., and Deng, Z.-H. (2017a). Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189.

Shen, X., Su, H., Li, Y., Li, W., Niu, S., Zhao, Y., Aizawa, A., and Long, G. (2017b). A conditional variational framework for dialog generation. In *Proceedings of the 55th ACL Conference on the Association for Computational Linguistics*, pages 504–509.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Tan, M., Xiang, B., and Zhou, B. (2016). Lstm-based deep learning models for non-factoid answer selection. In *Proceedings of the 4th ICLR International Conference on Learning Representations (Workshop track)*.

Wan, S., Lan, Y., Guo, J., Xu, J., Pang, L., and Cheng, X. (2016). A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2835–2841.

Wang, S. and Jiang, J. (2017). A compare-aggregate model for matching text sequences. In *Proceedings of the 5th ICLR International Conference on Learning Representations*.

Wang, M., Smith, N. A., and Mitamura, T. (2007). What is the jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of ACL-EMNLP Conference on Empirical Methods on Natural Language Processing*, pages 22–32.

Wang, B., Liu, K., and Zhao, J. (2016). Inner attention based recurrent neural networks for answer selection. pages 1288–1297. in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics.

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

Xiong, C., Zhong, V., and Socher, R. (2017). Dynamic coattention networks for question answering. In *Proceeding of the 5th ICLR International Conference on Learning Representations*.

Yang, Y., Yih, W.-t., and Meek, C. (2015). Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of ACL-EMNLP Conference on Empirical Methods on Natural Language Processing*, pages 2013–2018.

Yin, W., Schütze, H., Xiang, B., and Zhou, B. (2016). Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *TACL Transactions of the Association for Computational Linguistics*, 4:259–272.

Zhou, G., Zhou, Y., He, T., and Wu, W. (2016). Learning semantic representation with neural networks for community question answering retrieval. *Knowledge-Based Systems*, 93:75–83.