Building Interactive Sentence-aware Representation based on Generative Language Model for Community Question Answering

DOI:
10.1016/j.neucom.2019.12.107

Citation for published version (APA):
Wu, J., Mu, T., Thiyagalingam, J., & Goulermas, J. Y. (2020). Building Interactive Sentence-aware Representation based on Generative Language Model for Community Question Answering. *Neurocomputing*. https://doi.org/10.1016/j.neucom.2019.12.107

Published in:
Neurocomputing

Building Interactive Sentence-aware Representation based on Generative Language Model for Community Question Answering

Jinmeng Wu[a,d], Tingting Mu[b], Jeyarajan Thiyagalingam[c], John Y. Goulermas[a]

[a] *The school of Electrical Engineering, Electronics and Computer Science, The University of Liverpool, Brownlow Hill, Liverpool, UK*
[b] *The school of Computer Science, The University of Manchester, Kilburn Building, Oxford Road, Manchester, UK*
[c] *Science and Technologies Facilities Council, Rutherford Appleton Laboratory, Harwell Campus, Oxon, UK*
[d] *School of Electrical and Information Engineering, Wuhan Institute of Technology, China*

Abstract

Semantic matching between question and answer sentences involves recognizing whether a candidate answer is relevant to a particular input question. Given the fact that such matching does not examine a question or an answer individually, context information outside the sentence should be considered equally important to the within-sentence syntactic context. This motivates us to design a new question-answer matching model, built upon a cross-sentence, context-aware, bi-directional long short-term memory architecture. The interactive attention mechanisms are proposed which automatically select salient positional sentence representations, that contribute more significantly towards the relevance between two question and answer. A new quantity called context information jump is proposed to facilitate the formulation of the attention weights, and is computed via the joint states of adjacent words. An interactive-aware sentence representation is constructed by connecting a combination of multiple sentence positional representations to each hidden representation state. In the experiments, the proposed method is compared with existed models, using four public

Email addresses: sgjwu2@liv.ac.uk (Jinmeng Wu), tingting.mu@manchester.ac.uk (Tingting Mu), t.jeyan@stfc.ac.uk (Jeyarajan Thiyagalingam), j.y.goulermas@liverpool.ac.uk (John Y. Goulermas)

community datasets, and the evaluations show that it is very competitive. In particular, it offers 0.32%-1.8% improvement over the best performing model for three out of four datasets, while for the remaining one performance is around 0.2% of the best performer.

Keywords: Community questions answering; semantic matching; representation learning; recurrent neural network; attention mechanism

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 question answering (cQA) [1], whereas enabling the computer to automatically generate a novel answer, through some natural language model, is known as machine dialogue [2, 3]. In this work, 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 [4, 5], machine translation [6, 7, 8], and image caption generation [9, 10].

In order to compute an accurate measure of relevance between the sentence pair, 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 that improve a given similarity metric [11]. 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 [4, 12, 13].

As previously mentioned, the goal of a QA matching task is to select the correct answers from a set of candidate answers based on the content of a given question. Traditional works [12, 14] have built the neural networks based model to learn independent sentence representation in a sequential manner for computing the similarity score in the matching layer. However, the sentence representation is not enough robust for QA matching. The neural networks directly match the question and answer representations without involving word-to-sentence, sentence-to-sentence and un-ordered word-to-word interactions. Thus, learning the high-level word and sentence representations become a challenging task, thus the three motivations of building the proposed model are: 1) to learn different representations of the word in a complex scene such as polysemy, 2) to share key vocabulary components and semantic information between question and answer texts, and 3) to explore the relationship between positional words within a sentence.

Recently, pre-trained language models are widely used to improve QA matching performance [15, 16]. It is easy to observe that the polysemous word contains multiple meanings in different sentence contexts. For instance, the same word "apple" between the two sentences "Does Jobs like apple company, which he founded?" and "His favorite food is apple". Basic language models [17, 18] encode the learned single word embedding into entire contexts, which may lead to inaccurate sentence representation without considering the effect of word-to-sentence. The pre-trained language model focuses on generating the varied word representation for the corresponding context. In contrast to the existing pre-trained language models [15, 16], the proposed generative model enables to learn the word/sentence-level representations on the specific corpus for a cheaper pre-training technique. In QA task, the pre-training mechanism is beneficial to understand the lexical and syntactic information of the sentence.

With respect to the sentence-to-sentence interaction, in some cases, ambiguous content in question or answer sentence may impede the interactive process. For instance, consider the question-answer scenario given in Fig. 1. Regards to the object "cat" of query, A_1 provides more distinct keywords in answer than A_2. When focusing on fixed keywords in the question text, such as "cat" and "where", both answers contain information that matches these keywords, e.g., "cat" and "in the park" in A_1, and also "cat" and "on the mat" in A_2. This simple keyword-based matching strategy, hence, becomes a limitation on machine-based decision making.

Question Q_1

- Where was the cat?

Candidate Answers: A_1 and A_2

- A_1: I saw the cat before, I think it is in the park now.
- A_2: It is left on the mat, in a room with a cat.

Figure 1: Example scenario 1 for QA based on key-word matching.

However, if the focus of the sentence can be varied according to the context of the other sentence, e.g., by paying more attention to “cat” in the Q_1, and “in a room” instead of “on the mat” in A_2, the machine can then judge the correct keywords in A_2. Hence, the question-answer matching process becomes more effective when the sentence representations for questions and answers are learned jointly, other than in isolation. Past research on cQA [19, 20] has shown that it is important to model the content interaction between the question and answer sentences to improve the performance of a QA system. This interactive learning has been exploited in the previous work [21] using a hybrid attention model that includes a bi-directional long short-term memory (LSTM) model and a convolutional neural network (CNN). The attention mechanism incorporates question sentence context to generate the answer representation based on word-level representations. However, such one-way attention propagation may lose the semantic information captured in the other sentence. In the proposed work, a bidirectional context-aware model is built for a cross-sentence interactive learning by joining both question and answer sentence contexts.

Consider another QA scenario shown in Fig. 2, where has two question examples, each with their own pool of answers. We highlight the key components for each of the answers in Figs. 3 and 4. 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

Question Q_2

- Where was the cat?

Candidate Answers: A_{21} and A_{22}

- A_{21}: The cat was sitting on a mat.
- A_{22}: We had a dog that was friendly to our cat.

Question Q_3

- What is the color of that cat?

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

- A_{31}: The cat was sitting on a mat.
- A_{32}: The cat that was sitting on the red mat.

Figure 2: Example scenario 2 with two different QA cases.

Answer	**Key Components**
A_{21}	The ***cat*** was sitting ***on a mat***
A_{22}	We had a dog that was friendly to our ***cat***

Figure 3: Key components of potential answers to the Question Q_2.

by an attention mechanism [7]. Although interaction between the question and answer sentences can be formulated as a similarity accumulation over word pairs parameterized by weight variables (e.g., [13, 22]), the resulting model can be inflexible. This is because, when converting the discovery of the content interaction between the question and answer sentences to an optimization of the weight variables, fixed contributing patterns of word positions for discriminating the matching question-answer pairs are assumed.

Answer	**Key Components**
A_{31}	The ***cat*** was sitting on a mat.
A_{32}	The ***cat*** that was sitting on the ***red*** mat.

Figure 4: Key components of potential answers to the Question Q_3.

Given different questions, it is natural for a human to pay attention to different parts of the answer sentence. For instance, when reading “a white cat is sitting on the tree”, we pay more attention to “white” knowing the question is “what is the colour of the cat”, while more attention to “on the tree” if the question is changed to “where is the cat”. In this example, there also exist words that are naturally less informative, e.g., “a” and “the” as compared to “white”, “cat” and “sitting”. And this is not affected by the question content. Therefore, high attention weights should be selectively assigned to more informative word positions in the answer. To automatically identify non-informative words in a sentence and take this into account in attention weight assignment, we propose a new quantity referred to as the context information jump indicator. It captures the informativeness of a word by representing the across joint representation between adjacent words based on pre-trained language model. Including the proposed quantity as part of input, the importance of a word position in an answer sentence is affected not only by the answer and question content that is relevant to the matching task, but also its own informativeness independent of the matching.

In this work, we aim at improving the modeling of the question-answer interaction in representation learning through investigating effective ways of modeling the involved input. In this section, we address the aspects that have been highlighted above, particularly on the interactive learning of the question and answer representations and attention mechanism design, and propose a novel approach to improve the standard of the response accuracy during the cQA process. 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 answer 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 question 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 interaction-based and sentence-based two attention parallel mechanisms for sentence representation learning, and augment our proposed approach to consider the relationship between adjacent words, instead of concatenating the word representations to formulate co-attention weights as in previous works [23, 24].

3. A new quantity in co-attention mechanism, referred to as the context information jump, is proposed to represent the aggregation representation between forward and backward states based on the bi-directional LSTM. Context jump is able to modify the question for every words in answer, vice versa.

4. We perform an exhaustive evaluation of the proposed approach using four community datasets, namely TREC, Yahoo! and StackEx(L) and WikiQA, and share our findings.

The remaining sections is organized as follows: In Section 2, we review the related work. In Section 3, we review the operation of LSTM. Then we discuss the proposed method in Section 4, explaining the generative bi-directional-interaction model with the context jump information. This is followed by a detailed discussion on the evaluation process in Section 5, and results from evaluation in Section 6. We finally conclude the work in Section 7.

2. Related Work

A wide variety of techniques has been proposed in the literature for handling the cQA problem. We divide and present them under the three categories below.

2.1. Conventional Approaches

Lexical matching is a traditional technique for detecting semantic similarity between text objects. For instance, [25] evaluates the string similarity between words, and [26] develops a feature-based system, computing the similarity distances between words using a variety of statistical methods. One of the main drawbacks of such techniques is that the similarity between synonyms cannot be well captured directly from the text [27, 28].

This synonym-specific problem, however, can be addressed through a number of methods. One approach is to pre-compute or pre-load word co-occurrence information based on one or more large text corpus, such as Wikipedia. Another method is to leverage word hierarchy information drawn from semantic networks, such as WordNet, as in [29, 25]. Characterizing each word with a vector and comparing the words through a well-defined similarity function, such as cosine similarity [30], can also handle this specific problem. A number of techniques exist for generating an embedding vector for a word. Popular methods include bag-of-words (BOW) representation based on the contextual words around the target word [31, 32], latent semantic analysis (LSA) [33], distributed word embeddings generated by a probabilistic neural language model [17, 34, 35], and Gaussian distribution embedding [36].

Once the similarities between words are established, the similarity between a pair of sentences can be derived based on element-wise comparison of words using techniques like the syntactic tree kernel [37, 38], the tree edit distance (TED) [39] and its multiple variations [40, 41]. These techniques return a similarity matrix [42] between two given sentences. Nevertheless, despite measuring the similarity, the similarity matrix may not reflect the syntactic or global structure of the sentences [4].

2.2. Neural Semantic Models

Deep neural networks have been proven to be effective for generating distributed embedding representations of text objects (e.g., words, phrases and sentences) and characterizing the latent relationships between them. In the context of cQA, they have been widely applied to handle a number of problems, such as identifying paraphrased sentences [4, 43], detecting shared meaning between sentences [5, 44], and for syntactic parsing to capture semantic relationship between phrases [45, 46].

The CNN-based approaches have been very successful in image representation learning and very popular in text representation learning. Assuming that each sentence is characterized by a set of word-embedding vectors stored in a sentence matrix, a CNN can be typically employed to compute a vector representation for the sentence from its input matrix. A similarity score can then be computed between a pair of sentence vectors, by, for instance, a tensor-based operation [47]. Notable variations of this type of CNN-based matching model include, but are not limited to, [48, 14]. In particular, the similarity score computed from a CNN-based sentence representation can be treated as intermediate feature and combined with sentence representations themselves before further processing [12]. That is, the sentence representations returned by CNN for each sentence are concatenated with the scalar similarity score. The concatenated vector is compressed to a dense vector of lower dimension by a fully connected neural network. Like an CNN to convolve sentence, an auto-encoder can be applied to learn the sentence representation from word embeddings [49]. In [50], a restricted Boltzmann machine (RBM) is used to combine bag-of-word features and non-textual features for a given sentence, prior to feeding the fused features to a classifier to decide the best possible answer. Except above approaches to learn sentence representation, the other method works on learning the word-level semantics between sentences, where CNN learns the distributed similarity representation from sentence embedding matrices [51, 52].

However, none of above approaches account for the order of the words in a sentence. In recent years, recurrent neural network (RNN) [53] have become a popular choice in processing natural language due to their effectiveness in modeling the word order information within a sentence. For instance, [7] uses a bi-directional RNN facilitated by an alignment model [9] to compute the sentence representation for machine translation. In QA related tasks, [13] characterizes the sentence by using a stacked bi-directional LSTM, and [54] uses a bi-directional LSTM. In [54], the multiple hidden representations returned by a bi-directional LSTM at different states are used to compute a similarity matrix between the question and answer sentences. It is a common choice that learns a sentence representation using RNN or the variations of RNN (e.g. LSTM [55], GRU [56]), [57] utilizes a bi-directional LSTM to learn the word position representation of each time step in a sentence. Recent work [15] has proposed the use of multi-layer bi-directional LSTMs model for pre-training to learn the contextual word representations, followed by the downstream tasks.

2.3. Attention Mechanisms for cQA

The attention mechanism, first proposed in [7] for the NMT task, enables a neural network to identify the salient components of a sentence. It tends to rely on a weighted sum of a set of component representations, where the attention weights control the contributions of the components. The softmax function is typically used to convert a set of importance scores to a set of positive attention weights that sum to unity. Different ways of designing attention mechanisms correspond to different strategies of defining the components and formulating their importance scores. In the proposed model, we refer to a function that is used to compute these importance scores as an attention function.

A typical way of incorporating an attention mechanism in an RNN- or LSTM-based cQA system, is to relate the different components to the different hidden states of the network, which correspond to the different word positions in a sentence. The final sentence representation can be expressed as a weighted sum of the hidden representations computed at these states. In [20], the importance score is formulated as a function of each hidden representation itself, and focuses solely on the contribution of the word position within the target sentence. In [21], the attention mechanism is applied to the answer sentences, where the importance score is computed from not only the hidden representation of the answer states, but also the question representation returned by a bi-directional LSTM. This results in an interactive attention mechanism between answers and questions. Similar strategies to [21] are also proposed in [19, 58]. More sophisticated attention mechanisms are developed by considering more factors that may affect the importance score. For instance, [59] takes into account the previous episode memory, while [60] considers the question topic and question type in cQA, as well as the question and answer interaction information.

Instead of using attention mechanism in the learned representations from specific network, an alternative way to set the attention mechanism is to examine the importance of the word pairs that appear in the given sentence pair. For instance, given a question sentence containing n words and an answer sentence containing m words, each element in the $n \times m$ attention weight matrix indicates how much a word pair contributes to the relevance of the two given sentences. The importance score of each word pair can be computed from their corresponding word embeddings [32] or the hidden representations at the corresponding word positions returned by an LSTM [23], through the use of Euclidean distance or dot product. [61] measures the semantic interactions of word pairs from similarity matrix between the encoded sentences representations, which come from bi-directional LSTM. A soft alignment representation is computed for each word in sentence using an attention mechanism in word-level similarity matrix.

Variations of attention mechanism can be developed in a bespoke manner to suit a specific task, for instance, by taking into account an external knowledge base [62], by implementing an attentive max-pooling operation for CNN [63, 64], or by joining the internal documents into given question using co-attention attention in MRC task [23], etc. Typically, [24] explores an sentence-aware word attention on each word position representation of a sentence before computing the RNN representations. Besides the CNN or RNN-based attention models, a recent auto-encoder with attention model [65] applies a hidden representation from the encoder to reconstruct sentence representations in the decoder for question retrieval. Recently, self-attention has emerged as an attention mechanism aimed at aligning the multiple positions of a sequence, which has been widely used in a variety of the related QA tasks, for instance, machine reading comprehension (MRC) [66, 67], NMT [68] and abstractive summarization [69].

For instance, [70] provides the fusion functions to combine self-attention and similarity matrix based attention to complete the related MRC task. In cQA, [71, 72] apply a multi-dimensional self-attention mechanism to question and answer embeddings, and an attention weight vector instead of a single attention scalar is computed to learn word-level alignment representation. In section 4.1, we extend the self-attention mechanism by involving more contextual information in cQA datasets.

3. Preliminaries

A commonly used strategy for selecting from a candidate answer pool a sentence that matches the given question, is to first compute the representations, e.g., in the form of vectors or matrices, for the question and answer sentences based on their word content. Similarity (or relevance confidence) scores between the question and the candidate answers are then computed using their corresponding representations, and the candidate with the highest score is selected.

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. An RNN-based language model learns a vector representation to encode the semantic and order information of the words in the sentence. This is typically expressed as

$$\boldsymbol{h}_t = f(\boldsymbol{w}_t, \boldsymbol{h}_{t-1}), \tag{1}$$

where the t-th word x_t corresponds to a hidden state at time step t, and $\boldsymbol{w}_t$ denotes a vector representation for encoding the semantics of the word x_t. The hidden representation vector $\boldsymbol{h}_t$ contains word context information accumulated up to the t-th word in the sentence. It is computed from the vector representation $\boldsymbol{w}_t$ of the current word and the previous accumulation $\boldsymbol{h}_{t-1}$. The different realizations of the activation function $f(\cdot)$ result in different types of RNNs. For instance, a classical RNN employs a standard linear operation with a sigmoid activation sig$(\cdot)$ to process the input $\boldsymbol{w}_t$ and $\boldsymbol{h}_{t-1}$. Differently, an LSTM uses a set of recurrent functions [73] by following defined as

$$\boldsymbol{i}_t = \operatorname{sig}\left(\mathbf{W}_{xi}\boldsymbol{w}_t + \mathbf{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i\right), \tag{2}$$

$$\boldsymbol{f}_t = \operatorname{sig}\left(\mathbf{W}_{xf}\boldsymbol{w}_t + \mathbf{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f\right), \tag{3}$$

$$\boldsymbol{o}_t = \operatorname{sig}\left(\mathbf{W}_{xo}\boldsymbol{w}_t + \mathbf{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o\right), \tag{4}$$

$$\boldsymbol{g}_t = \tanh\left(\mathbf{W}_{xc}\boldsymbol{w}_t + \mathbf{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g\right), \tag{5}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t, \tag{6}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{c}_t\right), \tag{7}$$

where $\odot$ denotes the Hadamard product. The word vector $\boldsymbol{w}_t$, as well as the weight matrices $\mathbf{W}$ and the bias vectors $\boldsymbol{b}$ with different subscript symbols, are the model variables to be optimized.

To enrich the sentence representation, a bi-directional LSTM architecture can be used [7]. Specifically, one LSTM is used to process the input sentence as a sequence of words in the forward direction, of which the computed hidden representation at the t-th word position is denoted by the vector $\boldsymbol{h}_{t,f}$ (all vectors in this manuscript are considered column ones). A different LSTM processes the input sentence in the reverse direction, and the learned hidden representation is denoted by $\boldsymbol{h}_{t,b}$. Combining both, an extended hidden sentence representation at each word position is given as $\boldsymbol{h}_t = \left[\boldsymbol{h}_{t,f}^\top, \boldsymbol{h}_{t,b}^\top\right]^\top$, and is referred to as the positional sentence representation at the t-th word [13]. Working with the two sets of sentence positional representations $\{\boldsymbol{h}_t^{(q)}\}_{t=1}^N$ and $\{\boldsymbol{h}_t^{(a)}\}_{t=1}^M$, various strategies [13, 21, 58] are developed to compute their similarity or relevance confidence scores (we use the indicator symbols "q" and "a" to distinguish a question sentence from an answer sentence). The model used in this proposed work is described in sub-section 4.4.

4. Proposed Method

To summarize, the proposed cQA system contains a cross-sentence context-aware bi-directional LSTM referred to as CABIN model illustrated in Fig.5. The

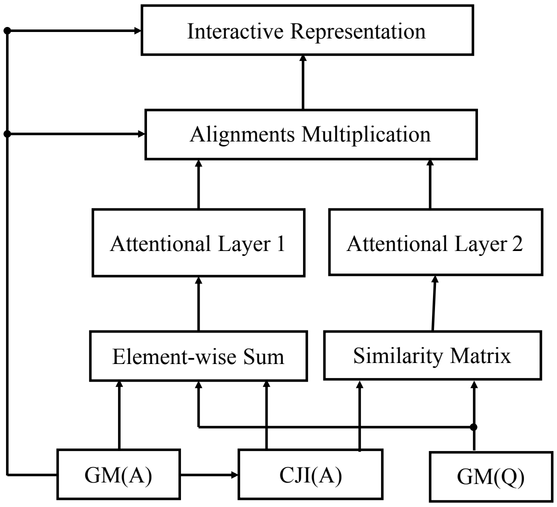


Figure 5: Architecture of the proposed CABIN system for computing interactive sentence representations. GM(A) symbol represents the pre-trained answer representation from the generative language model; GM(Q) symbol is the pre-trained question representation from the generative language model; CJI(A) symbol means the context information jump vector of the answer sentence.

proposed model is built upon an improved modeling strategy of the question-answer interaction, containing three key components: (1) the pre-trained language model benefits the proposed method, (2) the attention-driven interactive sentence-aware representation enhanced by context information jump, and (3) the distributed similarity computation. In the following sections, we describe the proposed system in detail.

4.1. Co-attention Sentences Mechanism

A common method [21] to formulate the self-attention function $A(\boldsymbol{h}_t, \mathbf{g})$ for each positional answer sentence representation is defined as

$$A(\boldsymbol{h}_t, \mathbf{g}) = \tanh\left(\boldsymbol{u}^T \boldsymbol{h}_t + \boldsymbol{v}^T \big(\tfrac{1}{T}\sum_{t=1}^{T} \boldsymbol{h}_t\big)\right), \tag{8}$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are the model parameters to be optimized. The sentence content is encoded by its averaged positional representations, given as $\mathbf{g} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{h}_t$.

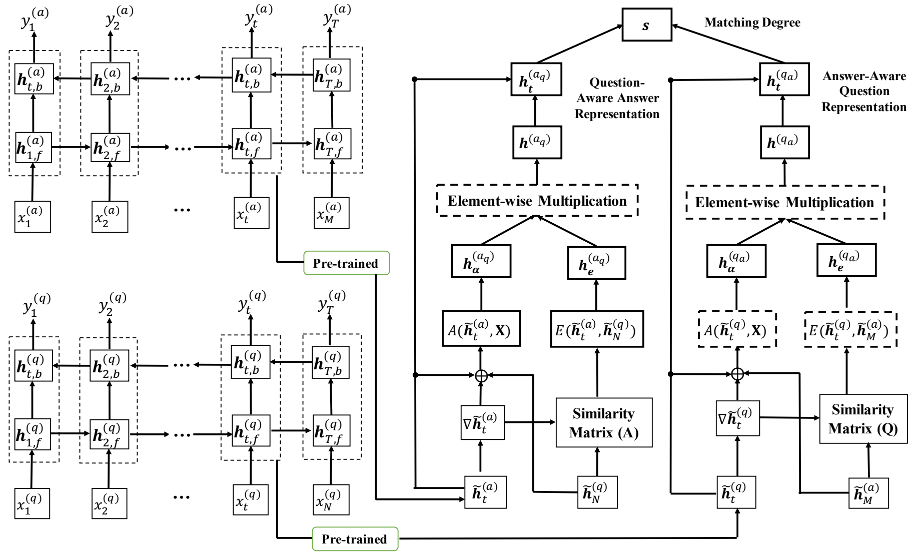

s
Matching Degree
$h_t^{(a_q)}$
Question-Aware Answer Representation
$h_t^{(q_a)}$
Answer-Aware Question Representation
$h^{(a_q)}$
$h^{(q_a)}$
Element-wise Multiplication
Element-wise Multiplication
$h_\alpha^{(a_q)}$
$h_e^{(a_q)}$
$h_\alpha^{(q_a)}$
$h_e^{(q_a)}$
$A(\tilde{h}_t^{(a)}, \mathbf{X})$
$E(\tilde{h}_t^{(a)}, \tilde{h}_N^{(q)})$
$A(\tilde{h}_t^{(q)}, \mathbf{X})$
$E(\tilde{h}_t^{(q)}, \tilde{h}_M^{(a)})$
$\nabla\tilde{h}_t^{(a)}$
Similarity Matrix (A)
$\nabla\tilde{h}_t^{(q)}$
Similarity Matrix (Q)
$\tilde{h}_t^{(a)}$
$\tilde{h}_N^{(q)}$
$\tilde{h}_t^{(q)}$
$\tilde{h}_M^{(a)}$
Pre-trained
Pre-trained
$y_1^{(a)}$ $y_2^{(a)}$ $y_t^{(a)}$ $y_T^{(a)}$
$h_{t,b}^{(a)}$ $h_{2,b}^{(a)}$ $h_{t,b}^{(a)}$ $h_{T,b}^{(a)}$
$h_{1,f}^{(a)}$ $h_{2,f}^{(a)}$ $h_{t,f}^{(a)}$ $h_{T,f}^{(a)}$
$x_1^{(a)}$ $x_2^{(a)}$ $x_t^{(a)}$ $x_M^{(a)}$
$y_1^{(q)}$ $y_2^{(q)}$ $y_t^{(q)}$ $y_T^{(q)}$
$h_{t,b}^{(q)}$ $h_{2,b}^{(q)}$ $h_{t,b}^{(q)}$ $h_{T,b}^{(q)}$
$h_{1,f}^{(q)}$ $h_{2,f}^{(q)}$ $h_{t,f}^{(q)}$ $h_{T,f}^{(q)}$
$x_1^{(q)}$ $x_2^{(q)}$ $x_t^{(q)}$ $x_N^{(q)}$

Figure 6: Architecture of the attention mechanisms for computing question-aware answer representations in the proposed CABIN system.

Each positional answer representation and the sentence content jointly control values of the attention weights.

In this work, we propose a new parallel and interactive attention mechanism with its architecture to compute the answer sentence representation as example illustrated in Fig.6. Here, we introduce the computation of the attention formulation $A(\cdot, \cdot)$, the attention formulation $E(\cdot, \cdot)$ would be represented in section 4.4. Assuming the length of question and answer sentences are defined as N, M, respectively. Two additional quantities $\tilde{\boldsymbol{h}}_N^{(q)}$ and $\nabla\tilde{\boldsymbol{h}}_t^{(a)}$ are included in context vector $\mathbf{c}_a$ when formulating the attention function of answer, given as

$$A(\tilde{\boldsymbol{h}}_t^{(a)}, \mathbf{c}_a) = \tanh\left(\boldsymbol{u}_a^T \tilde{\boldsymbol{h}}_t^{(a)} + \boldsymbol{v}_a^T \tilde{\boldsymbol{h}}_N^{(q)} + \boldsymbol{g}_a^T \nabla\tilde{\boldsymbol{h}}_t^{(a)}\right). \tag{9}$$

Two quantities $\tilde{\boldsymbol{h}}_M^{(a)}$ and $\nabla\tilde{\boldsymbol{h}}_t^{(q)}$ are used for formulating the attention of question

$$A(\tilde{\boldsymbol{h}}_t^{(q)}, \mathbf{c}_q) = \tanh\left(\boldsymbol{u}_q^T \tilde{\boldsymbol{h}}_t^{(q)} + \boldsymbol{v}_q^T \tilde{\boldsymbol{h}}_M^{(a)} + \boldsymbol{g}_q^T \nabla\tilde{\boldsymbol{h}}_t^{(q)}\right), \tag{10}$$

where final state representation vectors $\tilde{\boldsymbol{h}}_N^{(q)}$, $\tilde{\boldsymbol{h}}_M^{(a)}$ encode the content information of the question and answer sentence. Different from existing approaches with attention mechanism [23, 74], they are learned in an unsupervised way by following a sentence generation model. The pre-trained vectors $\tilde{\boldsymbol{h}}_M$ and $\tilde{\boldsymbol{h}}_N$ effectively reduce the computational complexity. Moreover, the probabilistic language model is an effective approach to encode semantic information carried by sentences. The vector $\nabla\tilde{\boldsymbol{h}}_t$ is the proposed jump quantity, and the vector $\boldsymbol{g}$ is the model variable associated with this quantity.

4.1.1. Generative Sentence Content Representation

Suppose the vector $\tilde{\boldsymbol{h}}_T$ corresponds to the final-state representation of a sentence, which is returned by pre-training a bi-directional LSTM. It is learned in an unsupervised way, by letting this LSTM operate as a generative model to solve a sentence generation task (here, we use the symbol "˜" to distinguish it from the notation $\boldsymbol{h}_T$ of Section 3, which also denotes the final-state representation vector of a question returned by a bi-directional LSTM, but trained in a supervised manner tailored to the cQA matching task). We now first describe the unsupervised training of $\tilde{\boldsymbol{h}}_T$ and then explain its advantages.

Taking a corpus containing question sentences only, a bi-directional LSTM is trained by maximizing the log-likelihood of generating these sentences. Following the probabilistic language model [17], we formulate the probability of generating a sentence $\boldsymbol{x} = \{x_1, x_2, \ldots, x_T\}$ as

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} \frac{\exp\Big(\mathbf{W}(x_t,:)\tilde{\boldsymbol{h}}_t + \boldsymbol{b}(x_t)\Big)}{\sum_{i=1}^{V} \exp\Big(\mathbf{W}(x_i,:)\tilde{\boldsymbol{h}}_t + \boldsymbol{b}(x_i)\Big)}, \tag{11}$$

where the weight matrix $\mathbf{W}$ and the bias vector $\boldsymbol{b}$ are the model variables to be optimized. The row number of $\mathbf{W}$ and the length of $\boldsymbol{b}$ are equal to the number of words in the question vocabulary list, the vocabulary size is V. The operations $\mathbf{W}(x,:)$ and $\boldsymbol{b}(x)$ extract the row in $\mathbf{W}$ and the element in $\boldsymbol{b}$ that correspond to the input word x. Stochastic gradient descent is used to optimize the model by following the same process as in [17].

The question representation $\tilde{\boldsymbol{h}}_N^{(q)}$ and the answer representation $\tilde{\boldsymbol{h}}_M^{(a)}$, computed separately from the answer and question representations, acts as a fixed

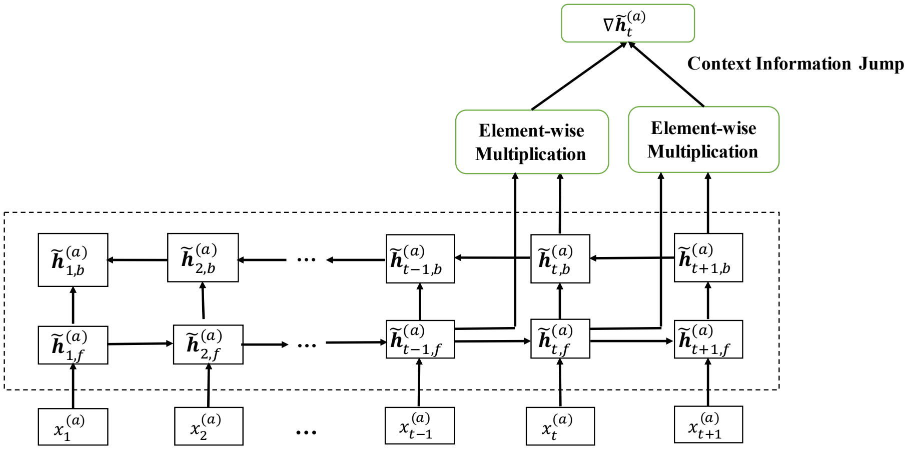

$\nabla\tilde{\boldsymbol{h}}_t^{(a)}$
Context Information Jump
Element-wise Multiplication
Element-wise Multiplication
$\tilde{\boldsymbol{h}}_{1,b}^{(a)}$
$\tilde{\boldsymbol{h}}_{2,b}^{(a)}$
…
$\tilde{\boldsymbol{h}}_{t-1,b}^{(a)}$
$\tilde{\boldsymbol{h}}_{t,b}^{(a)}$
$\tilde{\boldsymbol{h}}_{t+1,b}^{(a)}$
$\tilde{\boldsymbol{h}}_{1,f}^{(a)}$
$\tilde{\boldsymbol{h}}_{2,f}^{(a)}$
…
$\tilde{\boldsymbol{h}}_{t-1,f}^{(a)}$
$\tilde{\boldsymbol{h}}_{t,f}^{(a)}$
$\tilde{\boldsymbol{h}}_{t+1,f}^{(a)}$
$x_1^{(a)}$
$x_2^{(a)}$
…
$x_{t-1}^{(a)}$
$x_t^{(a)}$
$x_{t+1}^{(a)}$

Figure 7: Architecture of the bi-directional LSTM with context jump information in the proposed CABIN-LSTM system.

input to the attention function. As compared to Eq.(8) that requires simultaneous optimization of $\{\boldsymbol{h}_t\}_{t=1}^T$ together with $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{h}_t$, the pre-trained $\tilde{\boldsymbol{h}}_T$ effectively reduces the computational complexity. Moreover, the probabilistic language model is an effective approach to encode semantic information carried by sentences. The pre-trained question an answer representations are learned from bi-directional LSTM as the fixed input of the proposed matching model. We will show later in the result section that the proposed model offers competitive performance and the use of pre-trained $\tilde{\boldsymbol{h}}_T$ enhances the matching accuracy.

4.1.2. Context Information Jump

When learning sentence representations by a bi-directional LSTM, each obtained positional representation accumulates context information up to the targeted word position within a sentence in forward and backward directions. It is reasonable to assume that if the previous and next words bring significant change to the sentence semantics and content, 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 by the learned hidden representations between the current and adjacent states.

Therefore, given a sentence, we aim to formulate a quantity $\nabla\tilde{\boldsymbol{h}}_t$ that can be potentially used as an indicator of its information change between the current (t), the previous $(t-1)$ and the next $(t+1)$ word positions.

In the common technique of bi-directional LSTM, the positional word representation is affected by the neighboring word in a single direction during the propagation of bi-directional LSTM [7]. Here, we design a positional word state depends on the novel combination of current forward state and backward state. It is reasonable to assume the next state brings the context information to the current forward state. In a similar way, the previous state also enriches the current backward state. Thus, we explore the strategy to compute combined representation at current word position by involving the next state in backward direction and previous state in forward direction

$$\nabla\tilde{\boldsymbol{h}}_t^{(a)} = \begin{bmatrix} \tilde{\boldsymbol{h}}_{t,f}^{(a)} \odot \tilde{\boldsymbol{h}}_{t+1,b}^{(a)} \\ \tilde{\boldsymbol{h}}_{t,b}^{(a)} \odot \tilde{\boldsymbol{h}}_{t-1,f}^{(a)} \end{bmatrix}, \tag{12}$$

where $\odot$ is the Hadamard product [75], known as element wise product of two vectors. The matrix symbol $[:,:]$ aggregates the hidden states to a dimensional vector. We compute the alignment representation which is a good indicator of similarity between question and answer sentences. Because the quantity $\nabla\tilde{\boldsymbol{h}}_t^{(a)}$ of answer sentence is used as an indicator of the degree that new information is conveyed by the previous and next word between two adjacent states of an answer sentence, we refer to it as context information jump. Fig.7 illustrates the working operation of context information jump. Its role is to relate the salience of an sentence word position to the informativeness of this word given its adjacent ancestor word. By computing this quantity using a generative language model independent of the particular cQA task, general language patterns in sentence text can be captured. Using the same process for question sentence, we obtain the quantity $\nabla\tilde{\boldsymbol{h}}_t^{(q)}$.

4.2. Positional Word-Sentence Level Similarity

It is known that the semantic relativeness is a key component to determine the similarity between the question and answer sentence. In [61], they computed the similarity matrix between two sentences and applied it to compute the attention alignment representation. Inspired by this work, we design an adaptive similarity matrix to explore the importance of positional word in answer/question sentence for corresponding question/answer sentence. To achieve this, we compute the similarity between the positional word in answer and the question sentence, and vice versa. Specifically, we use pre-trained bi-directional LSTM model to solve the same sentence generation task as in Section 4.1.1. This results in a set of learned positional representations for the sentence, denoted by $\{\tilde{\boldsymbol{h}}_t\}_{t=1}^T$. By treating the final state of question sentence and the current state of answer sentence as the inputs into matching function $E(\tilde{\boldsymbol{h}}_t^{(a)}, \tilde{\boldsymbol{h}}_N^{(q)})$, given as

$$E(\tilde{\boldsymbol{h}}_t^{(a)}, \tilde{\boldsymbol{h}}_N^{(q)}) = \tanh\left(\boldsymbol{q}_1^{(a)}(\nabla\tilde{\boldsymbol{h}}_t^{(a)}(\tilde{\boldsymbol{h}}_N^{(q)})^T)\right), \tag{13}$$

where the weight $\boldsymbol{q}_1^{(a)}$ is a vector. The vectors $\tilde{\boldsymbol{h}}_{t,f}^{(a)}, \tilde{\boldsymbol{h}}_{t,b}^{(a)}$ indicate the pre-identifying hidden representation in t-th word in the forward and backward direction, separately. The output of matching function is a similarity vector representing the contextual relation between each word in target answer sentence and the question sentence. We employ the matching function to define the similarity-based attention weighted value $\boldsymbol{e}_t^{(a)}$ of question-aware answer as

$$e_t^{(a_q)} = \frac{\exp\left(\boldsymbol{q}_2^{(a)} E(\tilde{\boldsymbol{h}}_t^{(a)}, \tilde{\boldsymbol{h}}_N^{(q)})^T\right)}{\sum_{i=1}^T \exp\left(\boldsymbol{q}_{\boldsymbol{2}}^{(a)} E(\tilde{\boldsymbol{h}}_i^{(a)}, \tilde{\boldsymbol{h}}_N^{(q)})^T\right)}. \tag{14}$$

The variable vector $\boldsymbol{q}_2^{(a)}$ transfers the similarity vector to a matching score. The attention weight is computed based on the content from the similarity matrix between the answer word and question sentence. In addition to the attention weight in Eq.(15), this weight is also used to compute the answer representation in the next section. In a similar way, the similarity-based attention weight of answer-aware question $e_t^{(q_a)}$ could be computed.

4.3. Interactive sentence Representation

A method for modeling the interaction between two sentences is through the co-attention mechanism [13]. It utilizes a weight function to quantify the importance of the hidden sentence representation at the word position t. By incorporating the proposed attention formulation of Eq.(9), an importance weight between 0 and 1 is learned for each positional representation of the answer sentence $\tilde{\boldsymbol{h}}_t^{(a)}$, given as

$$\alpha_t^{(a_q)} = \frac{\exp\left(A(\tilde{\boldsymbol{h}}_t^{(a)}, \mathbf{c}_a)\right)}{\sum_{i=1}^{T} \exp\left(A(\tilde{\boldsymbol{h}}_i^{(a)}, \mathbf{c}_a)\right)}, \tag{15}$$

where $\mathbf{c}_a$ stores the question and adjacent words information that affects the importance of the targeted word position, and attention function $A(\cdot,\cdot)$ is computed in Eq.(9). Because the attention weight is affected by the question content and the importance of answer word, we adopt the notations of $\alpha_t^{(a_q)}$ and $e_t^{(a_q)}$ for each weight separately. The following alignment representation vectors are used to compute the two types of answer representation

$$\boldsymbol{h}_\alpha^{(a_q)} = \sum_{t=1}^{M} \alpha_t^{(a_q)} \tilde{\boldsymbol{h}}_M^{(a)}, \tag{16}$$

with

$$\boldsymbol{h}_e^{(a_q)} = \sum_{t=1}^{M} e_t^{(a_q)} \tilde{\boldsymbol{h}}_M^{(a)}. \tag{17}$$

This parallel weighted formulations encode information carried by each positional answer representation, and is weighted by an importance score that is affected by the question content, and also the positional representation and the word informativeness at the targeted word position. By combining these two alignment vectors, we compute the fused attention representation of answer sentence by

$$\boldsymbol{h}^{(a_q)} = \boldsymbol{h}_\alpha^{(a_q)} \odot \boldsymbol{h}_e^{(a_q)}. \tag{18}$$

To compute an adaptive answer sentence representation to the question content, we aggregate the pre-trained positional representation of answer sentence and the weighted representation, is defined as

$$\boldsymbol{h}_t^{(a_q)} = \tanh\left(\mathbf{V}_a(\tilde{\boldsymbol{h}}_t^{(a)} \odot \boldsymbol{h}^{(a_q)}) + \boldsymbol{b}_a\right), \tag{19}$$

where the weight matrices $\mathbf{V}_a$ and the bias vector $\boldsymbol{b}_a$ are model variables to be optimized. We denote each positional question representation computed with this modified architecture as $\boldsymbol{h}_t^{(a_q)}$, where $t = 1, 2, \dots, M$. The averaged alignment vector representation is used as the final state representation $\boldsymbol{h}_M^{(a_q)} = \frac{1}{M}\sum_{t=1}^{M} \boldsymbol{h}_t^{(a_q)}$, which refers to question-aware answer representation vector. The answer-aware question representation $\boldsymbol{h}_N^{(q_a)} = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{h}_t^{(q_a)}$, where the combined state $\boldsymbol{h}_t^{(q_a)}$ is computed from the formulation Eq.(19) for question.

4.4. Model Training and Initialization

So far, we have explained the computation of the question-aware answer representation vector $\boldsymbol{h}_T^{(a_q)}$ and the answer-aware question representation vector $\boldsymbol{h}_T^{(q_a)}$. Taking these two vectors as input, we formulate the following similarity vector to encode the distributed matching degree between the question and answer sentences

$$\boldsymbol{s} = \tanh\left(\mathbf{U}_q \boldsymbol{h}_N^{(q_a)} + \mathbf{U}_a \boldsymbol{h}_M^{(a_q)} + \boldsymbol{b}_s\right), \tag{20}$$

where the weight matrices $\mathbf{U}_q$, $\mathbf{U}_a$ and bias vector $\boldsymbol{b}_s$ are the model variables to be optimized. Subsequently, the sentence matching task can be formulated as a binary classification problem. The label $y = 1$ indicates that the answer is related to the question, while $y = 0$ otherwise. The probability that an answer is related to a question can be modeled using a two-way softmax function,

$$p(y = 1|\boldsymbol{s}) = \frac{\exp\left(\boldsymbol{s}^T \boldsymbol{\alpha}_1\right)}{\exp\left(\boldsymbol{s}^T \boldsymbol{\alpha}_0\right) + \exp\left(\boldsymbol{s}^T \boldsymbol{\alpha}_1\right)}, \tag{21}$$

where the two column vectors $\boldsymbol{\alpha}_0$ and $\boldsymbol{\alpha}_1$ are softmax parameters with the same dimensionality as $\boldsymbol{s}$. Based on the above formulation, model variables can be optimized by minimizing a regularized cross-entropy cost by following the logistic regression model [76, 77].

Here, we summarize the training process of the system. First, unsupervised pre-training of two individual bi-directional LSTM models are performed using the question sentences and answer sentences separately. Both models are trained to solve the language generation task via log-likelihood maximization, based on the sentence generation probabilities as formulated by Eq.(11). Sentence representations learned by these two models, e.g., $\tilde{\boldsymbol{h}}_N^{(q)}$ and $\{\tilde{\boldsymbol{h}}_t^{(a)}\}_{t=1}^M$, are used as the fixed input of the proposed matching model. Then, the matching model is trained to solve a binary classification problem by minimizing the regularized cross-entropy cost, based on the probability of observing a positive sentence pair as formulated in Eq.(21). Instead of random initialization, we initialize all the distributed word representation vectors with Glove word embeddings [18]. The bi-directional LSTM used for computing the question and answer representations are initialized by the pre-trained bi-directional LSTM model. The remaining variables are initialized randomly.

5. Evaluation Setup

In this section, we evaluate the proposed model CABIN against a number of state-of-the-art models using four key cQA datasets. Fristly, we present our evaluation methodology.

5.1. Datasets

We relied on four key cQA datasets for our evaluation, namely TREC[1] [78], Yahoo![2] [49], Stack-Exchange-Legal[3] (StackEx(L)), and WikiQA[4] [79]. We give a summary of the statistics related to these datasets in Table 1.

5.2. Performance Metrics

To report model performance using the test set, we use three performance metrics, namely mean reciprocal rank (MRR), mean average precision (MAP)

[1] http://trec.nist.gov/data/qa/t8qa_data.html

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

[3] https://law.stackexchange.com/

[4] https://aka.ms/WikiQA

Table 1: Dataset content statistics in CABIN model.

Parameter	**TREC**	**Yahoo!**	**StackEx(L)**	**WikiQA**
No. of Questions	1,505	90,000	6,939	3,047
No. of Answers	60,800	4.5M	8,595	29,258
Mean Question Length(words)	11.39	9.73	136.03	7.26
Mean Answer Length(words)	24.63	99.38	217.61	24.94

and the mean ranking of the top-N answers, denoted by MRT_N or p_N, as in [80]. The MRR metric focuses on the order of the correct answers, and is formulated as

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{r_i^1}, \tag{22}$$

where r_i^j denotes the computed ranking of the j-th correct answer in the ground truth ranking list for the i-th query, and $|Q|$ denotes the total number of queries tested. In other words, with $j = 1$, r_i^1 denotes the best possible answer. MAP accumulates the mean ranking of all the correct answers in each query, expressed as

$$\mathrm{MAP} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{n_i}\sum_{j=1}^{n_i}\frac{n_i^j}{r_i^j}, \tag{23}$$

where r_i^j is the computed ranking of the j-th correct answer from the ground truth ranking list for the i-th query, n_i^j is the number of truly correct answers in the computed ranking list of the j-th correct answer, and n_i denotes the number of truly correct answers for the i-th query.

5.3. Experimental Configuration

Experimental platform and recordings: All the training and testing were carried on a system with 36 physical cores, 128GB RAM, three graphical processing units (GPUS) each equipped with 12GB RAM, and running the version of the Tensor Flow Framework (v1.3).

Table 2: Benchmark data splits.

Data Set	**Q/A Pairs**	**Development**	**Training**	**Testing**
TREC [48]	8,997	1,148	4,718	1,517
Yahoo! [49]	4M	2,500	50,000	25,000
StackEx(L) [81]	7,760	1,500	4,760	1,500
WikiQA [79]	29,258	2,733	20,360	6,165

Neural network configurations: The bi-directional LSTM architecture used in our studies contains 100-dimensional hidden sentence representations. The dimensionality of each word embedding vector is set as 300.

Training preparation and initialization: In preparing the dataset for training and testing, we followed the same text pre-processing procedures described in [12]. More specifically, 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$\rangle$. Wherever the sentence lengths fall below the minimum threshold, a special symbol, $\langle$_PAD$\rangle$, is added to the end of the sentence, so as to pad them with extra characters to meet the processing requirements. Furthermore, the basic pre-training model is Glove [18] using a corpus containing 6B words from Wikipedia and Gigaword. For words appearing in each dataset, but not in their training corpus, a random value uniformly sampled from the interval of $[-0.3, 0.3]$ is assigned to each embedding dimension. A normal distribution $\mathcal{N}(0, 0.1)$ is used for model variables initialization.

Pre-trained language model configurations: The number of bidirectional LSTM layers is set to 2 with 100-dimensional hidden state size. For pre-trained language model optimization, we used the stochastic gradient decent without momentum with learning rate of 0.1, with a batch of 50 training examples for the gradient, with the gradient clipping norm threshold of 5. The learning rate is halved after 5 epochs using the polynomial decay function [82].

Training / testing process: For CANIN model optimization, a root mean square propagation (RMSProp) algorithm is used. For process includes a minibatch containing 50 training examples, a learning rate of 0.1, and a dropout rate of 0.5 [83]. The learning rate is halved after 10 epochs. Gradient clipping [84] is used to scale the gradient when the norm of gradient exceeds a threshold of five. The overall datasets have been split for training, testing and development purposes as suggested by the original datasets [48, 49, 81, 79], given in Table 5.2.

5.4. Baselines

To compare with the proposed method, the following ten models, stemming from the space of CNN, RNN and conventional/traditional techniques, are considered.

Baseline Models:

1. Random Guess (RandomGuess) [13]: A random ranking list for the test samples without training process.

2. Bag of Words (BoW) [51]: Each sequence of words is represented by the idf-weighted sum of the embeddings of the words it contains, and concatenated before feeding them as input to the network; for instance a multilayer perceptron (MLP).

3. Word Embedding (WordEmbed) [30]: This model uses the Glove tool to obtain the word embedding representation of a sentence. The matching score of two short-texts are calculated with an MLP, taking the embeddings of the two sentences as input.

CNN-based Models:

4. Bigram-CNN [48]: This model produces a sentence representation by feeding the adjacent words to a convolution layer, and then measures the similarity of the generated sentence representations through an MLP.

5. Add-CNN [12]: The model is an enhanced version of the Bigram-CNN model. It uses CNNs to produce the representations individually, and then calculates the matching score with an MLP.

6. AP-CNN [64]: It convolves each word embedding representation of the sentences, the output matrices from the convolution layer use the max pooling function with attention mechanism to learn the sentence representations.

7. Ab-CNN [32]: The model matches the feature maps of phase-level based sentences from the convolution layer to generate an attention matrix. It learns the high-level sentence representations as inputs to the convolutional layer, which is used to calculate the matching similarity.

8. CAM [52]: A recent work proposes the model performs different comparison matching functions to match the sentences based on word-level, where the similarity outputs from the function are aggregated into a vector by a convolution layer. The convolved vector as the input into the final prediction layer to compute the matching score.

RNN-based Models:

9. QA-LSTM [21]: Given two sentences, they are encoded by a bi-directional LSTM with a word-to-word attention mechanism, where the output from the model is fed to a convolution layer for producing the sentence representation.

10. IARNN [24]: The model learns an answer sentence representation using an attention mechanism to involve a question hidden representation from an RNN network, which then generates a high-level answer sentence representation as the input to the RNN network.

11. BiMPM [57]: The model encodes two sentences with a bi-directional LSTM, the encoded output of a sentence match each hidden representation of the other sentence in two directions. The sequences of matching vectors are aggregated into a vector as an input to prediction layer.

12. IWAN [61]: The model builds an alignment layer based on a word-level similarity martix for computing attention weight of each word, where the

similarity martix is computed by the sentence encoded outputs from a bi-directional LSTM.

Pre-training based Models:

13. ELMo [15]: The pre-trained BiLSTM model generates contextualized word vectors based on different contexts. The concatenated hidden representations from the BiLSTM are connected as inputs into the bidirectional attention flow model [74].

14. BERT [16]: The model demonstrates the deep transformers for pre-training the bidirectional word representations, which are used in the matching layer followed by fine-tuning the parameters of the model.

For the purpose of evaluation, we collect the reported results from the published works of above mentioned models, wherever possible. Wherever this was not feasible, we implemented them to match with the reported specification and experimental evaluation of these models.

6. Results and Analysis

6.1. Quantitative Evaluation

6.1.1. Comparison with State of the Art Methods

We first compare the performance of our proposed approach against a number of techniques using the metrics mentioned in Section 5. Table 3 reports the MRR and MAP metrics for different models evaluated using the four datasets mentioned above. Overall, the proposed model CABIN performs best in most cases, and is always amongst the top three performing models. In Table 4, we summarize the model ranking, where, for instance, the best performing model possesses the ranking of 1.0, while the worst possesses the ranking of 16.0. For each model, we report its averaged ranking over the two measures for each dataset, and the last column of the table reports the final averaged ranking over all the datasets. It can be seen from Table 4 that the proposed model possesses the highest ranking among all the compared ones.

Table 3: 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!		StackEx	
Models	MRR	MAP	MRR	MAP	MRR	MAP	MRR	MAP
RandomGuess [13]	0.5731	0.4920	0.4620	0.4582	0.4173	0.3759	0.4897	0.4350
BoW [51]	0.6810	0.5842	0.5411	0.5330	0.5021	0.4735	0.5842	0.5362
WordEmbed [30]	0.7052	0.6091	0.5630	0.5521	0.5273	0.4911	0.6053	0.5578
Bigram-CNN [48]	0.7846	0.7113	0.6415	0.6311	0.5952	0.5730	0.6821	0.6491
Add-CNN [12]	0.8078	0.7459	0.6652	0.6520	0.6150	0.5722	0.7067	0.6525
QA-LSTM [21]	0.8322	0.7111	0.7045	0.6821	0.6468	0.6157	0.7409	0.7158
AP-CNN [64]	0.8511	0.7530	0.6957	0.6886	0.6489	0.6047	0.7325	0.6830
Ab-CNN [32]	0.8539	0.7741	0.7108	0.6921	0.6530	0.6325	0.7461	0.7205
KV-MemNNs [85]	0.8523	0.7857	0.7265	0.7069	0.6749	0.6431	0.7580	0.7365
IARNN [24]	0.8208	0.7369	0.7418	0.7341	0.6687	0.6275	0.7489	0.7175
BiMPM [57]	0.8750	0.8020	0.7310	0.718	0.6892	0.6353	0.7523	0.7240
IWAN [61]	**0.8890**	0.8220	0.7500	0.7330	0.7010	0.6521	0.7689	0.7341
CAM [52]	0.8659	0.8145	0.7545	0.7433	0.7035	0.6630	0.7852	0.7483
ELMo [15]	0.8810	0.8247	0.7430	0.7369	0.7163	0.6758	0.7942	0.7537
BERT [16]	0.8827	0.8263	0.7592	0.7457	0.7218	0.6792	**0.8045**	0.7641
CABIN	0.8845	**0.8375**	**0.7653**	**0.7520**	**0.7250**	**0.6825**	0.8024	**0.7656**
CABIN-J	0.8563	0.7925	0.7415	0.7242	0.7026	0.6510	0.7812	0.7341
CABIN-A	0.8450	0.7843	0.7323	0.7135	0.6937	0.6442	0.7684	0.7150
CABIN-P	0.8559	0.7962	0.7320	0.7150	0.6883	0.6320	0.7763	0.7212

Table 4: Averaged ranking of different models. The best results are highlighted in bold and the second best are underlined.

Models	TREC	Yahoo!	StackEx	WikiQA	Overall
RandomGuess [13]	16.0	16.0	16.0	16.0	16.0
BoW [51]	15.0	15.0	15.0	15.0	15.0
WordEmbed [30]	14.0	14.0	14.0	14.0	14.0
Bigram-CNN [48]	12.5	13.0	12.5	13.0	12.75
Add-CNN [12]	11.0	12.0	12.5	12.0	11.9
QA-LSTM [21]	11.5	10.5	10.5	10.0	10.7
AP-CNN [64]	9.0	10.5	10.5	11.0	10.3
Ab-CNN [32]	7.5	9.0	8.5	8.5	8.4
KV-MemNNs [85]	7.5	8.0	6.5	5.5	6.9
IARNN [24]	11.0	5.5	8.5	8.5	8.4
BiMPM [57]	5.5	7.0	6.5	7.0	6.5
IWAN [61]	2.5	5.5	5.0	5.5	4.6
CAM [52]	5.5	3.0	4.0	4.0	4.1
ELMo [15]	3.5	4.0	3.0	3.0	3.4
BERT [16]	2.5	2.0	2.0	1.5	2.0
CABIN (Proposed)	**1.5**	**1.0**	**1.0**	**1.5**	**1.3**

In the following, we make a number of more specific observations from Table 3:

- With respect to the MRR, where a higher value indicates better performance, the proposed approach outperforms all models when evaluated against the WikiQA and Yahoo! datasets. In particular, the proposed outperforms the next best performing model, which is BERT model, by 1.8%, 0.61% and 0.32% respectively, on TREC, WikiQA and Yahoo! datasets.

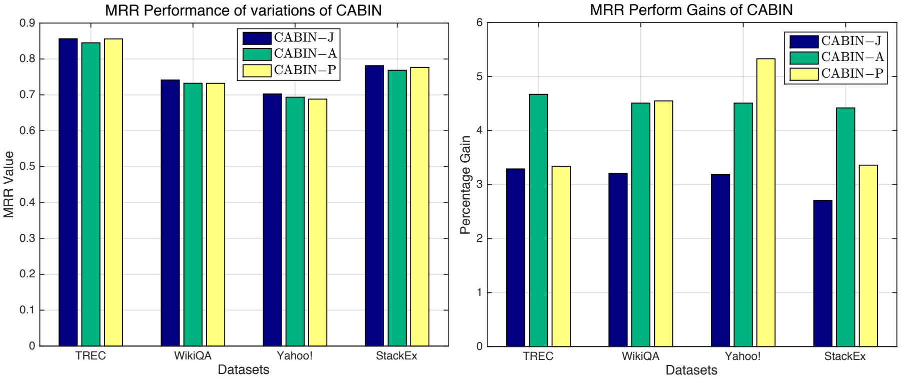
- When considering the MAP performance, the proposed approach outperforms the BERT model, when compared against the TREC, WikiQA, Yahoo! and StackEx(L) datasets, by 1.12%, 0.63%, 0.33%, and 0.15% respectively.

- On the TREC dataset, the proposed approach offers the best MAP performance, followed by the 2nd best IWAN model providing close performance. The proposed model beats the IWAN model by 1.55% in MAP performance.

- The MRR performance of the proposed model on the TREC and StackEx(L) datasets, however, are not as good as would be expected. The IWAN model achieves the best performance on on TREC dataset, which outperforms the proposed approach by 0.6%. For StackEx(L) dataset with a larger corpus, the BERT model achieves the better performance than the proposed model by 0.21%.

When comparing both MRR and MAP performance over the four datasets, the proposed model achieves the best results on TREC dataset, conversely, the worst results on Yahoo! dataset. Upon a closer inspection of the different datasets, we observe that there is a noticeable difference in mean lengths for questions and answers between the Yahoo! and other datasets. Also, the Yahoo! dataset contains questions and answers that are more informally formulated or expressed in a colloquial way, and this is particularly the case when compared against the TREC, StackEx(L) and WikiQA datasets. For example, in the Yahoo! dataset, it is common to see a question sentence like *"What Subbed episode does Nel transform???? @ Bobbi: Cause i saw it on youtube and yeah i just wanted to know, Thank you :-)"*, and a matching answer like *"Hmmm...bleach episode 192!!!!!!!!!! heres the list of the episodes lol:... GOOD LUCK!"*. Albeit being trivial, such informal formations of question-answer pairs render the cQA problem more difficult to handle.

In comparison, the three other datasets describe non-trivial, but well formulated question-answer pairs with long sentences. Both the proposed attention mechanism and the context information jump are developed to capture and encode information flow in sentences based on word semantics and order information. As such, the proposed model can better be exploited on the TREC, StackEx(L) and WikiQA datasets containing better formulated and longer question and answer sentences. Thus, it still has the challenge to solve the colloquial sentences matching in cQA datasets such as Yahoo! dataset.



(a) (b)

Figure 8: Left figure (a): Absolute performance and right figure (b): Performance gains of the proposed approach.

6.1.2. Empirical Analysis of CABIN model

To understand the performance behavior of our proposed CABIN in detail, and to verify the model varieties against our hypothesis, we trained and tested the proposed model under the three different conditions: without attention mechanism (CABIN-A), without context jump (CABIN-J), without pre-training process (CABIN-P). These evaluations enable the relative merits of the attention and context jump mechanisms and pre-training to be quantified over the proposed version. To assess the absolute advantage over the proposed version, we define the percentage gain on MRR as:

$$G_{MRR}(\mathbf{x}) = \frac{MRR_{(\text{CABIN})} - MRR_{\mathbf{x}}}{MRR_{\mathbf{x}}} \tag{24}$$

where $\mathbf{x} \in \{(\text{CABIN-A}), (\text{CABIN-J}), (\text{CABIN-P})\}$. Corresponding MRR performance and gains are shown in Figure 8. A number of observations can be made:

- When considering the absolute MRR performance (Figure 8(a)), the pre-training, the attention, and the context information jump mechanisms always improve a certain vaule in the MRR performance. Thus proposed model performance against the other varieties of the model.

- When considering the TREC and the StackEx(L) datasets (Figure 8(b)), the biggest contribution comes from the attention mechanism. The gain values of the two datasets are 4.67% and 4.42% among the proposed configurations.

- When considering the WikiQA and the Yahoo! datasets, the pre-training process contributs to the highest gain in MRR performance, by 4.55% and 5.33%, respectivity.

- The above observation is, for all datasets, where the jump mechanism produces a stable and similar gains around 3% in MRR performance. In particular, the jump mechanism brings the biggest increasement on TREC and WikiQA datasets, by 3.29% and 3.21%.

Overall, the above results show that the attention mechanism leads to a better performance to the highly contentable and structured type of the TREC and StackEx(L) datasets with less training samples. On the other hand, the preprocessing process is an efficient tool on the Yahoo! and WikiQA datasets with more training samples. The context jump mechanism considers the effect of adjacent text information leads to a better performance on highly structured and grammatically correct nature of the TREC and WikiQA datasets. This observation, to a certain extent, verifies the hypothesis that well-phrased English sentences are predictable.

6.2. Example Demonstration

To illustrate the efficacy of the proposed approach in a qualitative manner, we present a number of sample question-answer cases, from the best performing TREC and worst performing Yahoo! datasets. In both the cases, we show the top-three possible answers picked up by two different architectures. The two different, yet compatible, architectures are the IWAN model for TREC dataset, and CAM model for Yahoo! dataset, which are the second best performing models on the two datasets individually, hence chosen as a comparative model.

First consider the Example 1 and Example 2 from the TREC and Yahoo! datasets, presented in Tables 5 and 6. It can be observed that the true answers are correctly identified by the proposed model. Also, the ranked answers from the proposed model are more accurate than the ones from the IWAN model.

In addition to these two successful examples, we now consider an example where our model fails to handle an informally formulated question-answer pair from the Yahoo! dataset. For the question of *"ahh help, what is a really scary pea my pants scary story? I want it to be soo scary, Thank you :)"*, the correct available answer is *"oh god, man now that is really scary you – your pants from reading a scurry story lol... XD Hope it helps, X"*. However, when tested, the proposed model opted *"if you don't take a test, you'll continue to be scared. you should really just take it. just remember, if you're stressed and scared, your period can be late. it's best to just take a test to know for sure."* as the best answer, and selecting the ground truth answer as the second best answer. In this example, the ground truth answer contains informal language, the proposed model could not encode such information accurately and selects a longer sentence, which is more formally formulated, as the best answer.

To illustrate the efficiency of the proposed attention mechanism, we illustrate the salient word positions highlighted by the question-aware answer attention weights $\alpha_t^{(a_q)}$, for two example question-answer pairs from the TREC and Yahoo! datasets, in Table 7. Attention weights learned by the proposed and the existing attention mechanisms are reported for each pair. It can be seen from Table 7, that the proposed method is able to capture more accurately the salient word positions, which are important for the matching task.

To examine the efficiency of context jump mechanism in proposed model, we demonstrate corresponding similarities between an question and its context information jump using two example questions from the TREC datasetin, shown in Table 8. In the table, the word positions possessing the two largest context information jump and the two smallest context information jump are marked and indicated by T@k and B@k, respectively, for $k = 1, 2$. For each example question, a correct answer and an incorrect one are examined. It is interesting to observe that the T@k words are generally more informative than the B@k words.

For the same two example questions, we also illustrate the difference of the selected salient answer words and the top three retrieved answer sentences, between our two model versions CABIN-J and CABIN in Table 9. This is to demonstrate the effect of the proposed quantity of context information jump in attention learning and sentence matching. It can be seen from Table 9, that the inclusion of the proposed quantity results in more accurate answer retrieval and salient word identification for both example questions.

7. Conclusions

In this paper, we have proposed the cQA matching model CABIN, which is based on a cross-sentence context-aware bi-directional LSTM architecture. The goal is to improve the semantic matching between query and answer sentences, and this is achieved by exploring three aspects: contextual information between adjacent words in a sentence, an adaptive attention mechanism and the generative sentence representation by pre-processing based bi-directional LSTM. Thereby, we examine and analyze these specific skills benefit to cQA matching.

A novel pair-wise attention mechanism is designed to produce the interactive sentence representation, the first co-attention based on the sentence content, and the second interactive attention depends on the similarities between question and answer. In particular, we augment the existing techniques, which mainly use positional question and answer representations, with word frequency and co-occurrence information in order to improve the computation of attention weights. Further contributions of this work, include the context information jump and the use of a generative sentence representation. The former helps improving the computation of attention weights by considering informativeness of different word positions, whereas the latter eases the computation without sacrificing the representation quality.

Furthermore, to take into account adjacent context in sentence representation, the bi-directional LSTM learning representation is not only based on the simple previous or the next states in one direction propagation, but also on the use of the cross states of sentence in hand. This results in a context-aware inside the sentence representation, which is self-adaptive to the sentence content.

Overall, we evaluated the proposed model with the aid of four datasets, using a number of metrics and against a considerably large number of models from the literature including state-of-the-art ones. Our results indicate that the proposed attention mechanism, the proposed quantity of context information jump and the generated sentence representation can help to improve the question answer matching on a certain extent for different situations of datasets. Although further evaluations may be needed to differentiate the benefits on well-written text, our results indicate the proposed method is a very useful technique to improve the cQA process.

References

[1] C. Shah, J. Pomerantz, Evaluating and predicting answer quality in community QA, in: Proceedings of the 33rd ACM SIGIR International Conference on Research and Development in Information Retrieval, 2010, pp. 411–418.

[2] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, J. Pineau, Building end-to-end dialogue systems using generative hierarchical neural network models, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16), 2016.

[3] X. Shen, H. Su, Y. Li, W. Li, S. Niu, Y. Zhao, A. Aizawa, G. Long, A conditional variational framework for dialog generation, in: Proceedings of the 55th ACL Conference on the Association for Computational Linguistics, 2017, pp. 504–509.

[4] R. Socher, E. H. Huang, J. Pennin, C. D. Manning, A. Y. Ng, Dynamic pooling and unfolding recursive autoencoders for paraphrase detection, in:

Proceedings of the 24th NIPS Conference on Advances in Neural Information Processing Systems, 2011, pp. 801–809.

[5] J. Cheng, D. Kartsaklis, 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, 2015, pp. 1531–1542.

[6] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Proceedings of the 27th NIPS Conference on Advances in neural information processing systems, 2014, pp. 3104–3112.

[7] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: Proceedings the 6th ICLR International Conference on Learning Representations, 2015.

[8] J. Li, M. Luong, D. Jurafsky, A hierarchical neural autoencoder for paragraphs and documents, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 2015, pp. 1106–1115.

[9] A. Karpathy, L. Fei-Fei, Deep visual-semantic alignments for generating image descriptions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3128–3137.

[10] O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and tell: A neural image caption generator, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3156–3164.

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

[12] A. Severyn, A. Moschitti, 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, New York, NY, USA, 2015, pp. 373–382.

[13] S. Wan, Y. Lan, J. Guo, J. Xu, L. Pang, X. Cheng, A deep architecture for semantic matching with multiple positional sentence representations, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 2835–2841.

[14] X. Zhou, B. Hu, Q. Chen, X. Wang, Recurrent convolutional neural network for answer selection in community question answering, Elsevier, 2017.

[15] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, arXiv preprint arXiv:1802.05365.

[16] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805.

[17] Y. Bengio, R. Ducharme, P. Vincent, A neural probabilistic language model, Journal of Machine Learning Research 3 (2003) 1137–1155.

[18] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proceedings of ACL-EMNLP Conference on Empirical Methods in Natural Language Processing, Vol. 14, 2014, pp. 1532–43.

[19] W.-N. Hsu, Y. Zhang, J. Glass, Recurrent neural network encoder with attention for community question answering, arXiv preprint arXiv:1603.07044.

[20] H. Li, M. R. Min, Y. Ge, A. Kadav, A context-aware attention network for interactive question answering, in: Proceeding of the 23nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017.

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

[22] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, Q. V. Le, Qanet: Combining local convolution with global self-attention for reading comprehension, arXiv preprint arXiv:1804.09541.

[23] C. Xiong, V. Zhong, R. Socher, Dynamic coattention networks for question answering, in: Proceeding of the 5th ICLR International Conference on Learning Representations, 2017.

[24] B. Wang, K. Liu, J. Zhao, Inner attention based recurrent neural networks for answer selection, in: in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, 2016, pp. 1288–1297.

[25] A. Islam, D. Inkpen, Semantic similarity of short texts, Recent Advances in Natural Language Processing V 309 (2009) 227–236.

[26] D. Chen, J. Bolton, C. D. Manning, A thorough examination of the cnn/daily mail reading comprehension task, in: Proceedings of 54th ACL Annual Meeting of the Association for Computational Linguistics, 2016.

[27] L. Qiu, M.-Y. Kan, T.-S. Chua, Paraphrase recognition via dissimilarity significance classification, in: Proceedings of the 11th EMNLP Conference on Empirical Methods in Natural Language Processing, 2006, pp. 18–26.

[28] Z. Kozareva, A. Montoyo, Paraphrase identification on the basis of supervised machine learning techniques, in: Proceedings of the 19th NIPS Conference on Advances in Natural Language Processing, 2006, pp. 524–533.

[29] R. Mihalcea, C. Corley, C. Strapparava, Corpus-based and knowledge-based measures of text semantic similarity, in: Proceedings of the 20th AAAI Conference on Artificial Intelligence (AAAI-06), 2006, pp. 775–780.

[30] L. Kang, B. Hu, X. Wu, Q. Chen, Y. He, A short texts matching method using shallow features and deep features, in: Proceedings of the 3rd NLPCC Conference on Natural Language Processing and Chinese Computing, 2014, pp. 150–159.

[31] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv: 1301.3781.

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

[33] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, R. A. Harshman, Indexing by latent semantic analysis, Journal of the American Society for Information Science 41 (6) (1990) 391–407.

[34] A. Mnih, K. Kavukcuoglu, Learning word embeddings efficiently with noise-contrastive estimation, in: Proceedings of NIPS Conference on Advances in Neural Information Processing Systems, 2013, pp. 2265–2273.

[35] Y. Hao, T. Mu, R. Hong, M. Wang, X. Liu, J. Y. Goulermas, Cross-domain sentiment encoding through stochastic word embedding, IEEE Transactions on Knowledge and Data Engineering (2019) 1–1.

[36] L. Vilnis, A. McCallum, Word representations via gaussian embedding, in: Proceedings of the 4th ICLR Conference on International Conference on Learning Representations, 2015.

[37] A. Severyn, A. Moschitti, Automatic feature engineering for answer selection and extraction, in: Proceedings of the EMNLP Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (ACL), 2013, pp. 458–467.

[38] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, Building watson: An overview of the DeepQA project, AI Magazine 31 (3) (2010) 59–79.

[39] P. N. Klein, Computing the edit-distance between unrooted ordered trees, in: Proceedings of the 6th ESA Conference on Annual European Symposium on Algorithms, 1998, pp. 91–102.

[40] W.-t. Yih, M.-W. Chang, C. Meek, A. Pastusiak, Question answering using enhanced lexical semantic models, in: Proceedings of the 51st ACL Conference on Annual Meeting of the Association for Computational Linguistics, 2013, pp. 1744–1753.

[41] M. Wang, C. D. Manning, Probabilistic tree-edit models with structured latent variables for textual entailment and question answering, in: Proceedings of the 23rd COLING International Conference on Computational Linguistics, Association for Computational Linguistics (ACL), 2010, pp. 1164–1172.

[42] S. Fernando, M. Stevenson, A semantic similarity approach to paraphrase detection, in: Proceedings of the 11th CLUK Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics, 2008, pp. 45–52.

[43] W. Yin, H. Schütze, Convolutional neural network for paraphrase identification, in: Proceedings of the NAACL HLT Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 901–911.

[44] Z. Wang, H. Mi, A. Ittycheriah, Sentence similarity learning by lexical decomposition and composition, in: Proceeding of the 25th ACL-COLING International Conference on Computational Linguistics, 2016.

[45] R. Socher, C. D. Manning, A. Y. Ng, Learning continuous phrase representations and syntactic parsing with recursive neural networks, in: Proceedings of the NIPS Deep Learning and Unsupervised Feature Learning Workshop, 2010, pp. 1–9.

[46] T. Dozat, C. D. Manning, Deep biaffine attention for neural dependency parsing, in: Proceeding of the 5th ICLR International Conference on Learning Representations, 2017.

[47] X. Qiu, X. Huang, Convolutional neural tensor network architecture for community-based question answering, in: Proceedings of the 24th IJCAI International Joint Conference on Artificial Intelligence, 2015, pp. 1305–1311.

[48] L. Yu, K. M. Hermann, P. Blunsom, S. Pulman, Deep learning for answer sentence selection, in: Proceedings of the 27th NIPS deep learning workshop on Advances in Neural Information Processing Systems, 2014.

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

[50] H. Hu, B. Liu, B. Wang, M. Liu, X. Wang, Multimodal DBN for predicting High-quality answers in cQA portals, in: Proceedings of the 51th ACL Annual Meeting of the Association for Computational Linguistics, 2013, pp. 843–847.

[51] B. Hu, Z. Lu, H. Li, Q. Chen, Convolutional neural network architectures for matching natural language sentences, in: Proceedings of NIPS Conference on Advances in Neural Information Processing Systems, 2014, pp. 2042–2050.

[52] S. Wang, J. Jiang, A compare-aggregate model for matching text sequences, in: Proceedings of the 5th ICLR International Conference on Learning Representations, 2017.

[53] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, S. Khudanpur, Recurrent neural network based language model, in: Proceedings of Interspeech Conference on International Speech Communication Association, 2010, p. 3.

[54] D. Wang, E. Nyberg, A long short-term memory model for answer sentence selection in question answering, in: Proceedings of the 53th ACL Annual Meeting of the Association for Computational Linguistics, 2015, pp. 707–712.

[55] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.

[56] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, in: Proceedings of ACL-EMNLP Conference on Empirical Methods on Natural Language Processing, 2014.

[57] Z. Wang, W. Hamza, R. Florian, Bilateral multi-perspective matching for natural language sentences, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017.

[58] S. Romeo, et al., Neural attention for learning to rank questions in community question answering, in: Proceeding of the 26th ICLR Conference on International Conference on Computational Linguistics, 2016, pp. 1734–1745.

[59] C. Xiong, S. Merity, R. Socher, Dynamic memory networks for visual and textual question answering, in: Proceedings of the 33rd ICML Conference on International Conference on Machine Learning, 2016, pp. 2397–2406.

[60] X. Zhang, S. Li, L. Sha, H. Wang, Attentive interactive neural networks for answer selection in community question answering., in: Proceeding of the 31th AAAI Conference on Artificial Intelligence, 2017, pp. 3525–3531.

[61] G. Shen, Y. Yang, Z.-H. Deng, Inter-weighted alignment network for sentence pair modeling, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 1179–1189.

[62] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, J. Zhao, An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge, in: Proceedings of the 55th ACL Conference on Annual Meeting of the Association for Computational Linguistics, 2017, pp. 221–231.

[63] W. Yin, M. Yu, B. Xiang, B. Zhou, H. Schütze, Simple question answering by attentive convolutional neural network, in: Proceedings of the 26th ACL-COLING International Conference on Computational Linguistics, 2016.

[64] C. d. Santos, M. Tan, B. Xiang, B. Zhou, Attentive pooling networks, in: CoRR, Vol. abs/1602.03609, 2016.

[65] M. Zhang, Y. Wu, An unsupervised model with attention autoencoders for question retrieval, arXiv preprint arXiv:1803.03476.

[66] M. Hu, Y. Peng, Z. Huang, X. Qiu, F. Wei, M. Zhou, Reinforced mnemonic reader for machine reading comprehension, arXiv preprint arXiv:1705.02798.

[67] M. Hu, Y. Peng, Z. Huang, X. Qiu, F. Wei, M. Zhou, Reinforced mnemonic reader for machine reading comprehension, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), 2018.

[68] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[69] M. Yang, Q. Qu, Y. Shen, Q. Liu, W. Zhao, J. Zhu, Aspect and sentiment aware abstractive review summarization, in: Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 1110–1120.

[70] W. Wang, M. Yan, C. Wu, Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1, 2018, pp. 1705–1714.

[71] W. Wu, S. Xu, W. Houfeng, Question condensing networks for answer selection in community question answering, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1, 2018, pp. 1746–1755.

[72] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, C. Zhang, Disan: Directional self-attention network for rnn/cnn-free language understanding, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, 2018.

[73] H. Sak, A. Senior, F. Beaufays, Long short-term memory recurrent neural network architectures for large scale acoustic modeling, in: Proceedings of the 15th ISCA Annual Conference of the International Speech Communication Association, 2014.

[74] M. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi, Bidirectional attention flow for machine comprehension, in: Proceedings of the 5th ICLR International Conference on Learning Representations, 2017.

[75] G. P. Styan, Hadamard products and multivariate statistical analysis, Linear algebra and its applications 6 (1973) 217–240.

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

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

[78] M. Wang, N. A. Smith, T. Mitamura, What is the jeopardy model? a quasi-synchronous grammar for QA, in: Proceedings of ACL-EMNLP Conference on Empirical Methods on Natural Language Processing, 2007, pp. 22–32.

[79] Y. Yang, W.-t. Yih, C. Meek, Wikiqa: A challenge dataset for open-domain question answering, in: Proceedings of ACL-EMNLP Conference on Empirical Methods on Natural Language Processing, 2015, pp. 2013–2018.

[80] R. Baeza-Yates, B. Ribeiro-Neto, Modern information retrieval, Vol. 463, ACM press New York, 1999.

[81] A. Anderson, D. Huttenlocher, J. Kleinberg, J. Leskovec, Discovering value from community activity on focused question answering sites: a case study

of stack overflow, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 850–858.

[82] A. Borichev, Y. Tomilov, Optimal polynomial decay of functions and operator semigroups, Mathematische Annalen 347 (2) (2010) 455–478.

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

[84] R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks., in: Proceedings of the 30th ICML International Conference on Machine Learning, Vol. 28, 2013, pp. 1310–1318.

[85] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, J. Weston, Key-value memory networks for directly reading documents, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 2016, pp. 1400–1409.

Table 5: Comparison of the top three answers returned by the proposed CABIN and existing IWAN [61] architectures for an example question from the TREC dataset, where the ground truth answer sentences are marked by (∗) in the end.

Example 1 (TREC Dataset)	
Question	Who is the president or chief executive of Amtrak?
Top 3 answers by CABIN	**No.1**: “ long-term success here has to do with doing it right, getting it right and increasing market share,” said george warrington, amtrak ’s president and chief executive. (∗)
	No.2: “ amtrak is committed to treating all employees fairly,” amtrak president george warrington said in a statement. (∗)
	No.3: amtrak is also upgrading the tracks between washington and boston, said warrington, which should lead to improved service even before the high-speed trains are introduced.
Top 3 answers by IWAN [61]	**No.1**: amtrak will lose money again this year, but will meet the congressional deadline of weaning itself from operating subsidies by the fiscal year ending sept. 30 , 2002, officials said.
	No.2: “ amtrak is committed to treating all employees fairly,” amtrak president george warrington said in a statement. (∗)
	No.3: amtrak is offering a deal it hopes few travelers can resist: get good service or a free ride.

Table 6: Comparison of the top three answers returned by the proposed CABIN and existing CAM [52] architectures for an example question from the Yahoo! dataset, where the ground truth answer sentences are marked by (∗) in the end.

Example 2 (Yahoo! Dataset)	
Question	how to push yourself to the limit during excercising?
Top 3 answers by CABIN	**No.1**: try wearing a bandana and looking really cool and maybe you can "push yourself to the limit" in a top gun kind of way. listen to some bon jovy music. (∗)
	No.2: the answer to your question is no not necessarily. you probably are suffering from a subluxation of the lumabr spine.
	No.3: you have to break in a composite bat, which is what rolling it does. it's just like hitting a few hundred times. it works just fine, but the bats pop will probably die out sooner. but you will hit the ball harder and further.
Top 3 answers by CAM [52]	**No.1**: the red one is a shiny one meaning its rarer if i were you i would go for the red. but blue is good too, that the only difference is its colour.
	No.2: "squidward you like crabby patties don't you!?"
	No.3: try wearing a bandana and looking really cool and maybe you can "push yourself to the limit" in a top gun kind of way. listen to some bon jovy music. (∗)

Table 7: Comparison of the top three salient word positions in answer captured by the proposed CABIN and the second best models using two examples from the TREC and Yahoo! datasets. The learned attention weight is reported in parenthesis for each selected salient word.

Example 1 (TREC Dataset)	
Question	Who is the president or chief executive of Amtrak?
CABIN	“long-term success here has to do with doing it right , getting it right and increasing market share , ” said ***george*** (0.0751) warrington, ***amtrak***’s (0.0825) president and ***chief*** (0.0613) executive.
IWAN[61]	***amtrak*** (0.0612) will lose money again this year, but will ***meet*** (0.0469) the congressional deadline of weaning itself from operating subsidies by the fiscal year ending sept. 30, 2002, ***officials*** (0.0625) said.
Example 2 (Yahoo! Dataset)	
Question	how to push yourself to the limit during excercising?
CABIN	try wearing a bandana and looking really cool and maybe you can “***push*** (0.0754) yourself to the ***limit*** (0.0627)” in a top gun kind of way. ***listen*** (0.0516) to some bon jovy music.
CAM[52]	the red one is a shiny one meaning its rarer if i were ***you*** (0.0632) i would ***go*** (0.0562) for the red. but blue is ***good*** (0.0415) too, that the only difference is its colour.

Table 8: Illustration of answer word positions with either the largest two similarity values of the context information jump $\nabla\tilde{\boldsymbol{h}}_t^{(a)}$ indicated by T@K for K=1,2 (highlighted in bold), or the smallest two similarity values of $\nabla\tilde{\boldsymbol{h}}_t^{(a)}$ indicated by B@K for K=1,2 (underlined). We use **Q**, $\mathbf{A}_+$ and $\mathbf{A}_-$ to distinguish the question, correct answer and incorrect answer sentences.

Example 1	
Q: what is eileen marie collins' occupation ?	
$\mathbf{A}_+$: selected *by* (B@1) nasa *in* (B@2) January 1990, ***collins*** (T@1) became an ***astronaut*** (T@2) in July 1991.	$\mathbf{A}_-$: also, is she by any chance *from* (B@1) the ***daughter*** (T@2) *of* (B@2) michael collins, one of the ***apollo*** (T@1) 11 astronauts?
Example 2	
Q: what is the religious affiliation of the kurds ?	
$\mathbf{A}_+$: most ***kurds*** (T@2) are secular muslims who belong *to* (B@1) *the* (B@2) main ***sunni*** (T@1) sect.	$\mathbf{A}_-$: about 2 million kurds ***live*** (T@1) in northeastern syria near its border with ***turkey*** (T@2) and iraq, but *the* (B@1) kurdish military presence there centered mainly *around* (B@2) kurds from iraq, not turkey.

Table 9: Comparison of the top three answers and salient word positions returned by the two versions of CABIN-J and CABIN corresponding to ones with and without using the context information jump. The same two example questions as in Table 8 are examined, where the ground truth answer sentences are marked by (∗) in the end.

Example 1	
Question	what is eileen marie collins' occupation ?
Top 3 answers by CABIN	**No.1**: ***selected*** by ***nasa*** in January 1990, ***collins*** became an ***astronaut*** in July 1991. (∗)
	No.2: the five-member ***crew*** of the shuttle ***columbia*** that will launch chandra is led by veteran astronaut eileen ***collins*** , who would become the first woman of any nation to command a ***spaceflight***. (∗)
	No.3: also, is she by any ***chance*** from the ***daughter*** of michael ***collins***, one of the ***apollo*** 11 astronauts?
Top 3 answers by CABIN-J	**No.1**: also, is ***she*** by any chance from the daughter of michael ***collins***, one of the ***apollo*** 11 ***astronauts***?
	No.2: the five-member crew of the shuttle ***columbia*** that will launch chandra is led by veteran ***astronaut*** eileen ***collins***, who would become the first woman of any nation to command a ***spaceflight***. (∗)
	No.3: ***selected*** by nasa in January 1990, ***collins became*** an astronaut in ***July*** 1991. (∗)
Example 2	
Question	what is the religious affiliation of the kurds ?
Top 3 answers by CABIN	**No.1**: most ***kurds*** are secular ***muslims*** who ***belong*** to the main ***sunni*** sect. (∗)
	No.2: now his capture ***gives ocalan*** the ***stature*** among other ***kurds*** he never had before.
	No.3: about 2 million kurds ***live*** in northeastern syria near its ***border*** with ***turkey*** and iraq, but the kurdish military presence there centered mainly around ***kurds*** from iraq, not turkey.
Top 3 answers by CABIN-J	**No.1**: about 2 million kurds live in northeastern syria near its ***border*** with ***turkey*** and iraq, but the kurdish military presence there centered mainly around ***kurds*** from ***iraq***, not turkey.
	No.2: most ***kurds*** are secular ***muslims*** who ***belong*** to the ***main*** sunni sect. (∗)
	No.3: now his ***capture*** gives ***ocalan*** the ***stature*** among other ***kurds*** he never had before.