Learning English and Arabic Question Similarity with Siamese Neural Networks in Community Question Answering services

Nouha Othman, Rim Faiz, Kamel Smaïli

Learning English and Arabic Question Similarity with Siamese Neural Networks in Community Question Answering services

Nouha Othman[a,b], Rim Faiz[c], Kamel Smaïli[a]
othman@loria..fr, rim.faiz@ihec.rnu.tn, smaili@loria.fr

[a]*LORIA, Campus Scientifique, University of Lorraine, Vandoeuvre-lès-Nancy, F-54600, France*
[b]*LARODEC, ISG Tunis, University of Tunis, Bardo, Tunisia*
[c]*LARODEC, IHEC Carthage, University of Carthage, Carthage Presidency, Tunisia*

Abstract

In this paper, we tackle the task of similar question retrieval (QR) which is essential for Commu- nity Question Answering (cQA) and aims to retrieve historical questions that are semantically equivalent to the new queries. Over time, with the sharp increase of community archives and the accumulation of duplicated questions, the QR problem has become increasingly challenging due to the shortness of the community questions as well as the word mismatch problem as users can formulate the same query using different wording. Although many efforts have been devoted to address this problem, existing methods mostly relied on supervised models which significantly depend on massive training data sets and manual feature engineering. Such methods are chiefly constrained by their specificities that ignore the word order and do not capture enough syntactic and semantic information in questions. In this paper, we rely on Neural Networks (NNs) which use a deep analysis of words and questions to take into consideration the semantics as well as the structure of questions to predict the semantic text similarity. We propose a deep learning approach based on a Siamese architecture with Long Short-Term Memory (LSTM) networks, augmented with an attention mechanism to let the model give different words different attention while modeling questions. We also explore the use of Convolutional Neural Networks (CNN) nested within the Siamese architecture to retrieve relevant questions. Different similarity mea- sures were tested to predict the semantic similarity between the the pairs of questions. To evaluate the proposed approach, we conducted experiments on large-scale datasets in English and Arabic.

Keywords: Community question answering, Question retrieval, Siamese, LSTM, CNN.

1. Introduction

Along with the meteoric rise in popularity of web 2.0, community Question Answering (cQA) sites have emerged as a viable method for seeking information online. CQA sites such as

Yahoo! Answers[1], Stackoverflow[2], Quora[3], WikiAnswers[4], and Google Ejabat[5], give people the ability to post their various questions and get them answered by other users. Interestingly, users can directly obtain short and precise answers rather than a list of potentially relevant documents. Community sites are exponentially growing over time, building up very huge archives of previous questions and their answers. However, multiple questions with the same meaning can make information seekers spend more time searching for the best answer to their question. Therefore, retrieving similar questions could greatly improve the QA system and benefit the community. Detecting similar previous questions that best match a new user's query is a critical and challenging task in cQA, known as question retrieval. When good matches are detected, the answers to similar previous questions can be used to answer a new query. This could dodge the lag time incurred by waiting for new answers, thus enhancing user satisfaction. Owing to its importance, the question retrieval task has recently received wide attention [30, 3, 2, 26, 34, 32]. One big challenge for this task is the word mismatch between the new posted questions and the existing ones in the community archives [30]. Word mismatch means that similar questions can mean the same thing but they may differ lexically and syntactically. For instance, the questions *How can I protect myself from the Coronavirus?* and *What preventative measures should be taken to reduce the risk of getting COVID-19?* have nearly the same meaning but include different words and then may be regarded as dissimilar. This constitutes a barricade to traditional Information Retrieval (IR) models since users can phrase the same question using different wording. Furthermore, community questions have different lengths, mostly short and usually have sparse representations with little word overlap. Although numerous attempts have been made to tackle this problem, most existing methods focus on a unique domain and/or language and rely on the bag of-words (BOWs) representations which are constrained by their specificities that put aside the word order and ignore semantic and syntactic relationships. Recent advances in natural language and particularly question retrieval have been achieved using Neural Networks (NNs) [16, 24, 12, 11, 5] which perform a deep analysis of words and consider the semantics and the structure of questions in order to detect similar questions. In this paper, we propose an approach based on NNs to detect the semantic similarity between the questions. Neural networks are preferred as they generally tend to perform better than traditional

[1] http://answers.yahoo.com/
[2] http://stackoverflow.com/
[3] https://fr.quora.com/
[4] https://wiki.answers.com/
[5] https://ejaaba.com/

machine learning models. They also do not rely on linguistic features and consequently can be easily applied to languages other than English. The proposed approach is based on a Siamese architecture with LSTM networks, augmented with an attention mechanism. This latter is an additional layer, which make the model give different attention to different words while representing the questions in order to explicitly aggregate states with weights. The Siamese neural network architecture is well known for its ability to compute similarity requiring less training data. We also implement a Siamese CNN- based model to explore its efficiency on the question retrieval task. We tested different similarity measures to compare the final hidden states of the LSTM layers. Our experimental results have proven that our approach is significant for identifying relevant questions in English and Arabic.

2. Related Work

Recently, a whole host of methods have been proposed to address the question retrieval task. Early works were based on the vector space model referred to as VSM to calculate the cosine similarity between a query and archived questions [8, 3]. Nevertheless, the main limitation of VSM is that it favors short questions, while cQA services can handle a wide variety of questions not limited to factoïd questions. In order to overcome this shortcoming, Okapi BM25 (BM stands for Best Matching) has been used by search engines to estimate the relevance of questions to a given search query taking into account the question length [3]. Language Models (LM)s

[4] have been also used to model queries as sequences of terms instead of sets of terms. LMs estimate the relative likelihood for each possible successor term taking into account relative positions of terms. However, such models might not be effective when there are few common words between the questions. In order to handle the vocabulary mismatch problem faced by LMs, a model based on the concept of machine translation, referred in the following as translation model, was employed to learn correlation between words based on parallel corpora and it has obtained significant performance for question retrieval. The intuition behind translation-based models is to consider question-answer pairs as parallel texts then, relationships of words can be built by learning word-to-word translation probabilities like in [30, 2]. Within this context, Zhou et al. [36] attempted to enhance the word-based translation model by adding some contextual information when translating the phrases as a whole, instead of translating separate words. Singh et al. [26] extended the word-based translation model by integrating semantic information and explored strategies to learn the translation probabilities between words and concepts using the cQA archives and an entity catalog. Even though the above-mentioned basic models have yielded interesting results, questions and answers are not parallel in practice, rather they are different from the information they contain [34]. Further methods based on semantic similarity were proposed for question retrieval toward a deep understanding of short text to detect the equivalent questions. For instance, there have been a handful of works that have

exploited the available category information for question retrieval such as in [4, 3, 37]. Although these attempts have proven to improve the performance of the language model for question retrieval, the use of category information was restricted to the language model. Wang et al [28] used a parser to build syntactic trees of questions, and rank them based on the similarity between their syntactic trees. Nonetheless, such an approach requires a lot of training data and existing parsers are still not well-trained to parse informally written questions. Latent Semantic Indexing (LSI) was also used to address the given task like in [23]. Although LSI has proven to be effective in addressing the polysemy and synonymy by mapping terms related to the same concept close to each other, the efficiency of LSI depends on the data structure and both its training and inference are computationally expensive on large vocabularies. Other works focused on the representation learning for questions, relying on a Word Embedding for learning distributed representations of words in a low-dimensional vector space. As we believe that the representation of words is crucial for retrieving similar questions, we rely on word embeddings to represent the community questions. Along with the popularization of word embeddings and its capacity to produce distributed representations of words, advanced NN architectures such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) and LSTM have proven effectiveness in extracting higher-level features from the constituting word embeddings. For instance, Dos Santos et al. [7] employed CNN and bag-of-words (BOW) representations of the questions to calculate the cosine similarity scores. Within the same context, Mohtarami et al. [16] developed a bag-of- vectors approach and used CNN and attention-based LSTMs to capture the semantic similarity between the community questions and rank them accordingly. LSTM model was also used in [24] with an attention mechanism for capturing long dependencies in questions. Interestingly, the weights learned by the attention model were exploited for selecting important segments and enhancing syntactic tree-kernel models. More recently, the question retrieval task was modeled as a binary classification problem in [12] using a combination of LSTM and a contrastive loss function to effectively memorize the long term dependencies. In our work, we use a siamese adaptation of LSTM [17] for pairs of variable-length sentences named MaLSTM. This latter has accomplished excellent outcomes in the semantic text similarity task and inspire us in our question retrieval problem. It is worth noting that work on cQA has been mostly carried out for other languages than Arabic. The most promising approach [16] used text similarities at both sentence and word level based on word embeddings. The similarities were computed between new and previous question, and between the new question and the answer related to the previous question. A tree-kernel-based classifier was employed in [1] where the authors used supervised and unsupervised models that operated both at sentence and chunk levels for parse tree based representations. A supervised learning approach was adopted in [15] where learning-to-rank models were trained over word2vec features and covariance word embedding features produced from the training data. More recently, the given task was investigated by Romeo et al. [25] using advanced Arabic text representations made by applying tree kernels to

constituency parse trees along with word embeddings and textual similarities.

3. Description of the proposed ASLSTM approach

In order to improve the QR task, we propose an Attentive Siamese LSTM approach for question retrieval, referred to as ASLSTM to retrieve the semantically similar questions in cQA [21]. As illustrated in Figure 1, our approach is composed of three main modules namely, question preprocessing, word embedding learning and Manhattan LSTM (MaLSTM).

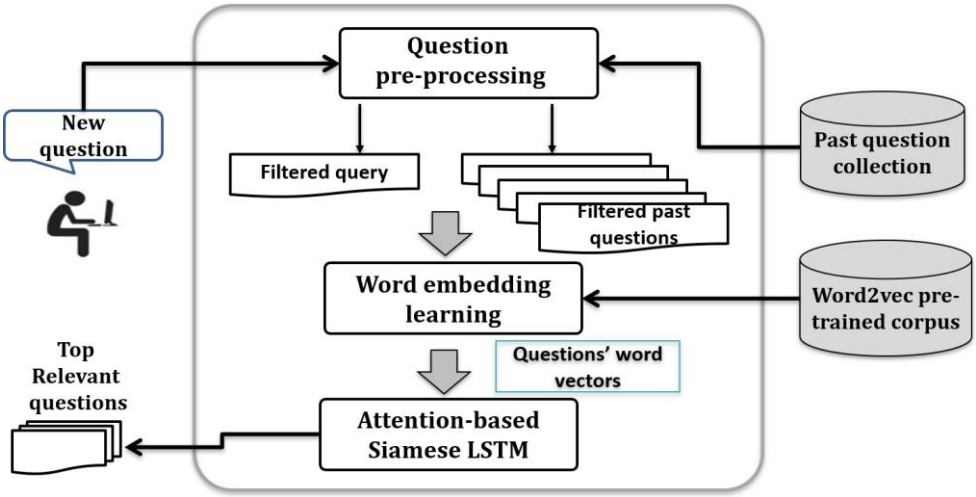


Figure 1: ASLSTM pipeline for question retrieval in cQA

The basic principle underlying the ASLSTM approach is to map every question word token into a fix-sized vector. The word embeddings of the questions are therefore fed to the Siamese LSTM with the aim of representing them in final hidden states encoding the semantic meaning of the questions. An attention mechanism is integrated in the Siamese architecture to determine which words should give more attention on than other words over the question. Community questions are then ranked by means of the Manhattan similarity function based on the vector representation of each question. A previous posted question is considered to be semantically equivalent to a queried question if their corresponding LSTM representations lie close to each other according to the Manhattan similarity measure. The historical question with the highest Manhattan score will be returned as the most similar question to the new posted one. The components of ASLSTM are detailed below.

3.1. Question preprocessing

Pre-processing is important to make the question collections cleaner and easier to

process. The question preprocessing module aims to filter the natural language community questions an extract the useful terms in order to represent them in a formal way. This module comprises text cleaning, tokenization, stopwords removal and stemming. Punctuation marks, non letters, diacritics, and special characters are removed. English letters are lowercased while dates are normalized to the token *date* and numerical digits are normalized to the token *num*. For the Arabic question collection, in addition to the aforementioned tasks, orthographic normalization was applied, including diacritics removal, stretching removal and Letter normalization.

3.2. Word Embedding Learning

Word embeddings are low-dimensional vector representations of words, learned by harness- ing large amounts of text corpora using shallow neural networks. The use of word embeddings allows to effectively detect the syntactic and semantic similarities between words. Particularly, we resorted to the Continuous Bag-of-Words (CBOW) model which has proven to outperform Skip gram on our datasets [19]. Recall that CBOW consists in estimating a pivot word according to its context using a window of contextual words around it, while Skip gram does the inverse predicting the contextual words given a current word in a sliding window. In our word embedding learning module, we map every word into a fix-sized vector using the Word2Vec model.

3.3. LSTM

Long Short-Term Memory (LSTM) [10], is a powerful type of RNN widely used in deep learning, and has proven its capacity to capture short and long-term dependencies and model sequential data. Interestingly, LSTM helps prevent the vanishing gradient problem [9] which is the main limitation of RNN. Gradient descent is an optimization algorithm that uses an iterative process to minimize a given function and improve the deep learning. The basic intuition is to adjust the weights of the model by determining the error function derivatives according to each member of the weight matrices in the model. To minimize the total loss, the gradient descent updates each weight in proportion to the derivative of the error with respect to that weight. LSTM is endowed with a memory cell that is capable of maintaining its state over time, and internal mechanisms called gates to regulate the information flow. The major reason for relying on LSTM in our approach is its proven performance in handling variable-length sequential data.

Given input vector xt, hidden sate ht and memory state ct, the updates in LSTM are performed as follows:

$$i_t = sigmoid(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$

$$f_t = sigmoid(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$

$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3)$$

$$c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \quad (4)$$

$$o_t = sigmoid(W_o x_t + U_0 h_{t-1} + b_0) \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

where i_t, f_t, o_t are input, forget, and output gates at time t, respectively. W_k, U_k are LSTM pa- rameterized weight matrices, b_k represents the bias vector for each k in $\{i, f, c, o\}$ and ($\odot$) denotes an element-wise product of matrices, known as the Hadamard product which is an entrywise multiplication.

3.4. Siamese Manhattan LSTM

The overall aim of this module is to compare a pair of sentences to decide whether or not they are semantically equivalent. We used the Siamese network [17] architecture which is known to have identical sub-networks LSTM left and LSTM right that are passed vector representations of two text sequences and return a hidden state encoding semantic meaning of the sequences. These hidden states are then compared using a similarity metric to return a similarity score as depicted in Figure 2.

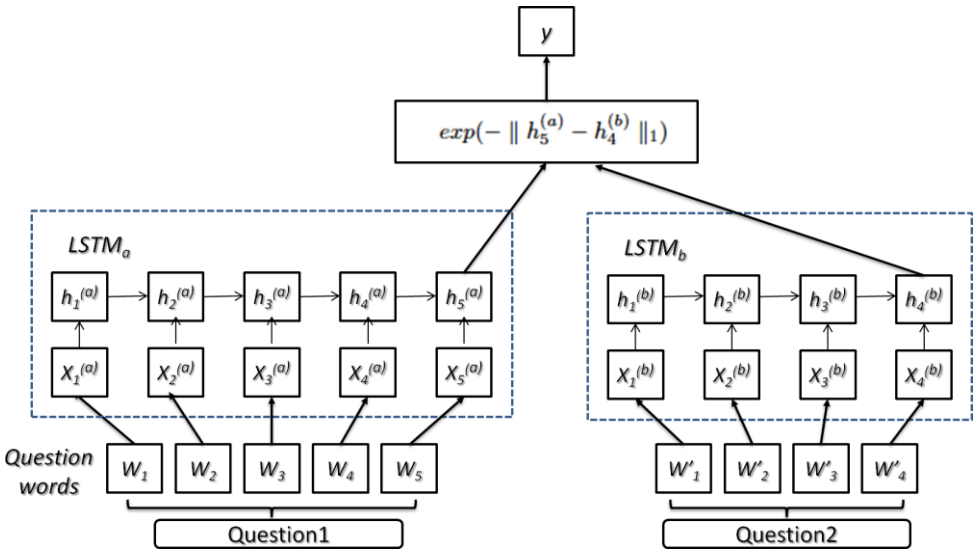


Figure 2: General architecture of the MaLSTM model

Note that we decided to use LSTM for each sub-network, but it is also possible to swap LSTM with GRU (Gated Recurrent Unit). GRU is a variation on the LSTM, also aiming to solve the vanishing gradient problem which comes with a standard RNN. GRU is almost similar to LSTM in terms of design, although they have two gates, namely reset gate and update gate. Reset gate determines how to combine new input with previous memory while update gate is what input gate and forget gate were in LSTM, determining how much of the previous state to keep Although GRU has less parameters and then might take less time to train, LSTM empirically remembers longer text sequences than GRU and usually outperforms them in tasks requiring modeling long-distance relations. This is the reason why we opted for LSTM rather than GRU.

In our work, Siamese LSTM was adapted to the context of question retrieval, that is

to say, the sentence pairs become pairs of questions.

LSTM learns a mapping from the space of variable length sequences din and encode the input sequences into a fixed dimension hidden state representation drep. More concretely, each question represented as a word vector sequence (e.g., Q1 is represented by x1, x2, x3) is fed into the LSTM, which updates, at each sequence-index, its hidden state. The final state of LSTM for each question is a drep-dimensional vector, denoted by h in figure 2, which holds the inherent semantic meaning of the question.

Once we have the two vectors that capture the underlying meaning of each question, the semantic similarity between the questions is computed using the following Manhattan similarity measure:

$$y = e^{-||h(left)-h(right)||1} \quad (7)$$

Note that since we have an exponent of a negative, the Manhattan function scores will be between 0 and 1. It is worth mentioning that we tested different similarity metrics, namely Manhattan, cosine and Euclidean distances and the best results were obtained with the Manhattan distance as will be seen later in the next section.

3.5. Attention Mechanism

Attention mechanism with Neural networks have recently achieved tremendous success in several NLP tasks [40, 6, 29]. We assume that every word in a question contributes to the meaning of the whole question but the words do not have equal influential information. Thus, we should assign a probability to every word to determine how influential it is to the entire question. Note that we adopted an attention mechanism as in [31]. Siamese LSTM model employs only the last hidden states of sequence pair e.g. h(a) and h(b), which may ignore some information. To remedy this problem, in our attention layer, we used all hidden states H = {h1, h2, ..., hT }, where hi is the hidden state of the LSTM at time step i summarizing all the information of the question up to xi and T denotes the length of the question. The general architecture of the proposed Siamese Manhattan LSTM model augmented with an attention layer is illustrated in Figure 3, where the different constituent layers are shown from the input (question words) to the output (similarity score).

Note that α(a) and α(b) denote the weights of LSTMa and LSTMb, respectively. Basically, the attention mechanism measures the importance of a word through a context vector uh. It computes a weight αi for each word annotation hi according to its importance. The final question representation r is the weighted sum of all the word annotations using the attention weights,

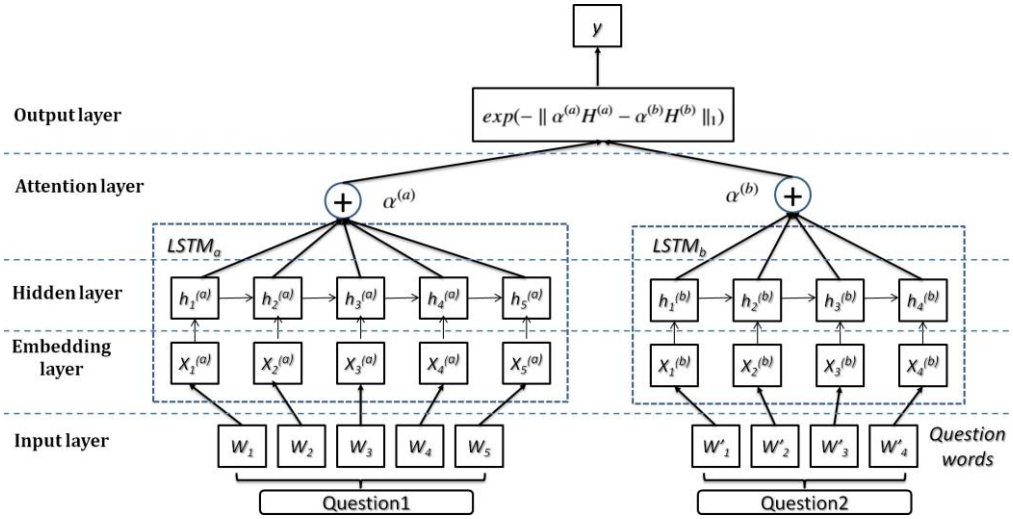


Figure 3: General architecture of our Siamese Manhattan LSTM model with attention mechanism

computed by equation 10. In the attention layer, a context vector u_h is introduced, which is randomly initialized and can be viewed as a fixed query, that allows to identify the informative words.

$$e_i = \tanh(W_h h_i + b_h) \quad with \quad e_i \in [-1, 1] \tag{8}$$

$$\alpha_i = \frac{\exp(e_i^T u_h)}{\sum_{i=1}^{T} \exp(e_t^T u_h)} \quad with \quad \sum_{i=1}^{T} \alpha_i = 1 \tag{9}$$

$$r = \sum_{i=1}^{T} \alpha_i h_i \quad with \quad r \in R^{2L} \tag{10}$$

where W_h, b_h , and u_h are the learnable parameters with W_h is a weight matrix and b_h is a bias vector used to project each context vector into a common dimensional space and L is the size of each LSTM.

4. Siamese Manhattan CNN

We also tested the Siamese architecture with CNN for the question retrieval task as depicted in Figure 4. CNN was widely used to model text sequences, taking as input the word embedding of words of text sequences aligned sequentially, and summarizing the meaning of a sequence through layers of convolution and pooling, until reaching a fixed length vector representation. CNN has the advantage of maintaining the word order information which is important for short sequences. Moreover, non linear activation in the convolutional neural networks can learn more latent characteristics about the text sequences. The Siamese CNN based approach is composed of two parallel identical convolutional structures CNN right and CNN left that receive vector representations of

two questions and each convolutional structure returns the question level representations. A similarity measure is then used to calculate the similarity between the pair of convolutional structures. Each convolutional structure includes a convolutional layer and max pooling layer and a fully connected layer. The convolutional layer is fed with a matrix, where each row is the vector representation of a word in the question. In the convolution layer, numer- ous filters slide over the rows of the matrices. The learned output vector from the fully connected layer will be then employed to compute the similarity using the Manhattan distance which has outperformed the cosine and Euclidean distances.

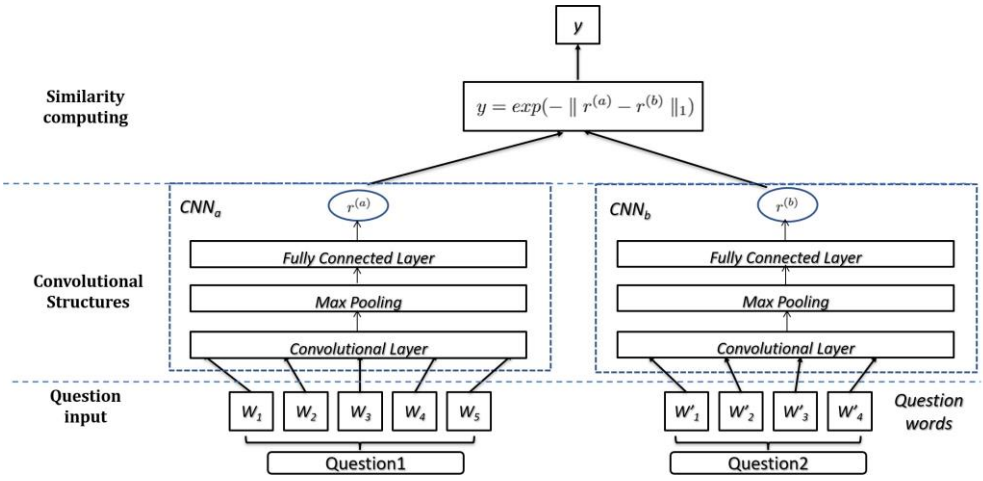


Figure 4: General architecture of the Siamese Manhattan CNN model

4.1. Convolution layer

The convolutional layer aims to extract patterns, such as discriminative word sequences in the input questions that are common throughout the training instances. This layer applies a set of *m* convolutional filters to a sliding window of width *w* over the matrix of previous embedding layer to extract new local w-gram features. The structure of convolution used for text classification [13] is shown in Figure 5. Let $F \in \Re^{w \times d}$ be a filter matrix. The result of each filter *F* will be a feature map denoted *f*, where the i-th element of *f* is learnt as follows:

$$f_i = fct(T_{i:i+w-1} \circ F + b) \quad (11)$$

Where fct denotes a non linear activation function, ◦ represents a convolution operation, Ti:i+w−1 denotes the token vectors ti, ti+1, , ti:i+w−1 (if k > N, tk= 0), and $b \in \Re$ is a bias term. We used the Rectified Linear Unit (ReLU) activation function [18] which is the most popu- lar activation function for neural networks, especially CNNs mainly owing to its representational sparsity, computational simplicity and linear behavior. It was chosen because it simplifies back- propagation, speeds up learning and avoids

saturation. Note that we applied multi-channel CNN using multiple convolving filters to extract active local n-gram features in different sizes. Token vectors are padded before convolution in order to keep identical size for outputs of different filters.

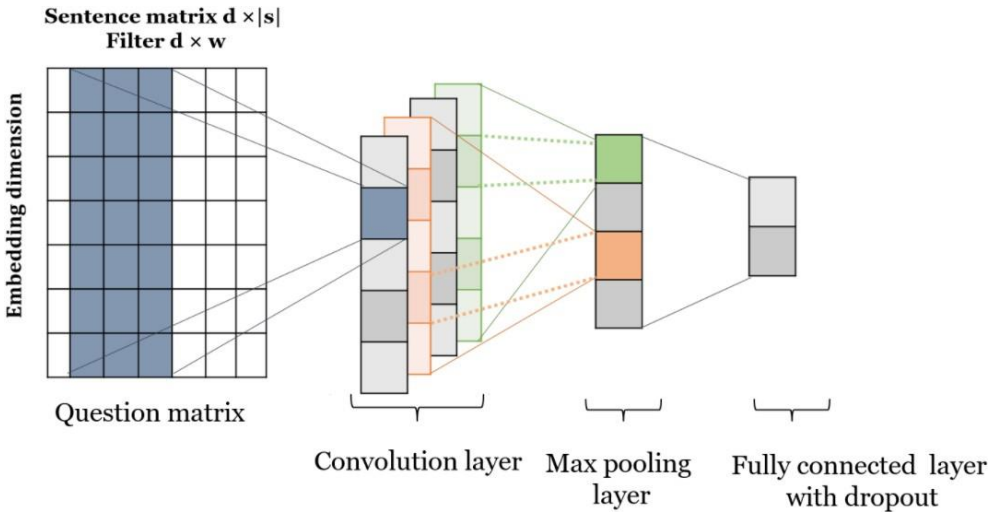


Figure 5: Convolutional Neural Networks for sentence classification [13]

4.2. Max-pooling and Dropout Layer

The pooling layer aggregates the vectors in the feature map matrix by taking the maximum value for each feature vector learned in the convolutional layer. This reduces the representation of both questions. We uses max rather than mean pooling as the salient feature represents the most distinguishing trait of a question. Max-pooling allows to reduce the computation for upper layers by eliminating non-maximal values. We applied a dropout layer [27] after both a convolution and max-pooling layer to prevent overfitting.

4.3. Fully-connected Layer

The fully connected layer is the terminal layer of the convolutional neural networks. It converts the output of the previous layers and the result of ReLU into a fixed-length semantic vector encoding the input to the network.

5. Experiments

5.1. Datasets

In order to evaluate the proposed approach, we performed experiments using the same dataset released by [35] for evaluation. The questions of the community collection were harvested from all categories in the popular Yahoo! Answers community platform, and then were randomly splitted into two sets while maintaining their distributions in all

categories. The first set is a question repository for question search containing 1,12M questions, while the second is the test set containing 252 queries and 1624 manually annotated related questions. Each original query in the test set, has an average of 15 candidate related questions. Note that 60% of these related questions are relevant to the queries and then annotated as positive. The community questions in the collection are in various structures, different lengths varying from 2 to 20 words as shown in Figures 6 and 7 and belonging to diverse categories e.g., Health, Sports, Computers and Internet, Diet and Fitness, Pets, Travel, Business and Finance, Entertainment and Music, Education and Reference, etc.

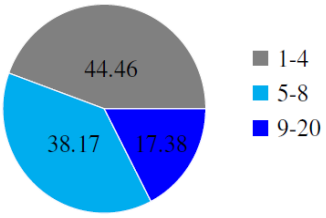


Figure 6: Distribution of questions' length in the English collection

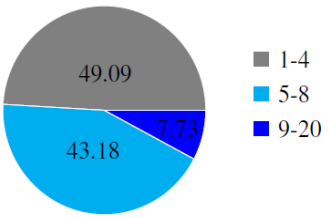


Figure 7: Distribution of questions' length in the Arabic collection

For our experiments in Arabic, we resorted to the same English collection translated using Google Translation with a careful manual verification, as there is no large Arabic dataset avail- able for the question retrieval task. The Arabic collection includes exactly the same number of questions as the English set. We randomly checked a number of questions and verified that the Google tranlation result is satisfactory for the question similarity task. It is worth mentioning that in the context of the question answering, even if the question is poorly translated, what matters most is that it keeps its semantics.Therefore, it is possible to answer a question even if it is poorly translated In order to train word2vec for Arabic, we resorted to a sizeable data set from cQA sites, namely the Yahoo!Webscope dataset[6], translated into Arabic using Google Translation, including 1 256 173 questions with 12 512 034 distinct words. For Siamese LSTM training, we employed the publicly available Quora Question Pairs dataset[7]. The given collection encompasses 400 000 samples of question duplicate pairs, where each sample has a pair of questions along with ground truth about their corresponding similarity (1: similar, 0: dissimilar). It is worth noting that data preprocessing was performed using Python NLTK.

5.2. Word Embedding Learning

For English word embedding training, we resorted to the publicly available word2vec[8]

[6] http://research.yahoo.com/Academic Relations
[7] www.kaggle.com/quora/question-pairs-dataset
[8] https://code.google.com/p/word2vec

vectors, with dimensionality of 300, that were trained on 100 billion words from Google News. Since no Arabic version of Google News vectors yet exists, we train the translated Yahoo!Webscope dataset using the CBOW model, as it has proven through experimentation to be more efficient and performs better than Skip-gram with our data. The training parameters of the CBOW model on the Arabic collection were set after several tests as follows:

- Size=300: feature vector dimension. Note that we tested different values in the range [50, 500] but did not get significant difference in terms of precision values. The best pre- cision was reached with size=300.
- Sample=1e-4: down sampling ratio for the words that are very redundant in the corpus.
- Negative samples=25: number of noise words
- min-count=1: minimum number of words which we set to 1 to make sure we do not throw away anything.
- Context window=5: fixed window size.

5.3. LSTM Training

During LSTM training, we applied the Adadelta method [33] for weights optimization to automatically decrease the learning rate. Gradient clipping was also used with a threshold value of 1.25 to avoid the exploding gradient problem [22]. Our LSTM layers' size is 50 and embedding layer's size is 300. We employed the back propagation and small batches of size equals 64, to reduce the cross-entropy loss and we resorted to the Mean Square Error (MSE) as a common regression loss function for prediction. We trained our model for several epochs to observe how the results varied with the epochs. We found out that the accuracy changed with the variation of the number of epochs but stabilized after epoch 25. The given parameters were set based on several empirical tests; each parameter was tuned separately on a development set to pick out the best one. For implementing our model we used Keras[9] and Scikit-learn[10]. Note that we used the same LSTM configuration for both languages.

5.4. CNN Training

The parameters of the CNN network were tuned on the validation dataset. The values of the parameters for which we obtained the best results are as follows: We ran a total of 25 epochs and the batch size of each epoch is 128. The Adam [14] optimizer was used as it is known to perform very well on deep neural networks. During the optimization, we set the learning rate to be 0.01. We set the number of Filters to 32, the kernel size of the three repeated convolutional layers are set as 3, Pool size to 2, Sliding window to 3 and Dropout to 0.5.

[9] https://keras.io/
[10] https://scikit-learn.org

5.5. Evaluation Metrics

To evaluate our approach, we used Mean Average Precision (MAP), Precision@n (P@n) and Recall as they are the most widely used metrics for assessing the performance of question retrieval in cQA. We remind that MAP assumes that the user is interested in finding many relevant questions for each query and then rewards methods that not only return relevant questions early, but also get good ranking of the results. Precision@n gives an idea about the classifier's ability of not labeling a positive sample as a negative one. It returns the proportion of the top-n retrieved questions that are equivalent. Recall is the measure by which we check how well the model is in finding all the positive samples of the dataset. It returns the proportion of relevant similar questions that have been retrieved over the total number of relevant questions. We also used Accuracy, which returns the proportion of correctly classified questions as relevant or irrelevant.

5.6. Results and Discussion

In order to test the performance of ASLSTM, we compare it against our previous approach called WEKOS as well as the competitive state-of-the-art question retrieval methods tested by Zhang et al. in [35] on the same dataset. The methods being compared are recalled below:

- **WEKOS** [20]: A word embedding based method which transforms words in each question into continuous vectors. The questions are clustered using Kmeans and the similarity between them was measured using cosine similarity based on their weighted continuous valued vectors.
- **TLM** [30]: A translation based language model which uses a translation-based lan- guage model with a query likelihood approach for the question and the answer parts respectively. TLM integrates word-to-word translation probabilities learned by using different sources of information.
- **ETLM** [26]: An entity based translation language model, which is an extension of TLM where the major difference is the replacement of the word translation with entity translation in order to integrate semantic information within the entities.
- **PBTM** [36]: A phrase based translation model which uses machine translation probabilities assuming that QR should be performed at the phrase level. PTLM learns the probability of translating a sequence of words in a historical question into another word sequence of words in a given query.
- **WKM** [39]: A world knowledge based model which integrates the knowledge of Wikipedia into the questions by deriving the concept relationships that allow to identify related topics between the queries and the previous questions. A concept thesaurus was built based on the semantic relations

extracted from Wikipedia.

- **M-NET** [38]: A continuous word embedding based model, which integrates the category information of the questions to get a category based word embedding, sup- posing that the representations of words belonging to the same category should be semantically equivalent.
- **ParaKCM** [35]: A key concept paraphrasing based approach which explores the translations of pivot languages and expands queries with the paraphrases. It assumes that paraphrases give additional semantic connection between the key concepts in the queried question and those of the historical ones.

Table 2 gives a comparison of the performance of ASLSTM against the aforementioned models on the English Yahoo! Answers dataset. The results in Table 2, show that PBTM outperforms TLM which demonstrates that detecting contextual information in modeling the translation of entire phrases or consecutive word sequences is more effective than translating separate words, as there is a dependency between adjacent words in a phrase.

Table 2: Question retrieval performance comparison of different models in English.

	TLM	ETLM	PBTM	WKM	M-NET	ParaKCM	WEKOS	ASLSTM
P@5	0.3238	0.3314	0.3318	0.3413	0.3686	0.3722	0.4338	**0.5033**
P@10	0.2548	0.2603	0.2603	0.2715	0.2848	0.2889	0.3647	**0.4198**
MAP	0.3957	0.4073	0.4095	0.4116	0.4507	0.4578	0.5036	**0.5799**

ETLM performs as well as PBTM, which proves that entity translation is more efficient than the word translation for ranking and could enhance the performance of the translation language model. WKM is based on Wikipedia as an external knowledge resource to derive the concept relationships, but its performance is limited by the low coverage of the Wikipedia concepts on the diverse users' questions. ParaKCM achieves good precision by exploring the translations of pivot languages and expanding queries with the produced paraphrases for question retrieval. M-NET, also based on word embeddings, performs well owing to the use of metadata of category information to derive the properties of words. WEKOS based on word embedding too along with TF-IDF weighting and kmeans, achieves comparable results and further proves that the use of word embeddings get benefits from dense word representation and mitigate the negative impact of word mis- match by capturing semantic relations between words, while the other methods mostly do not capture enough information about the semantic similarity.

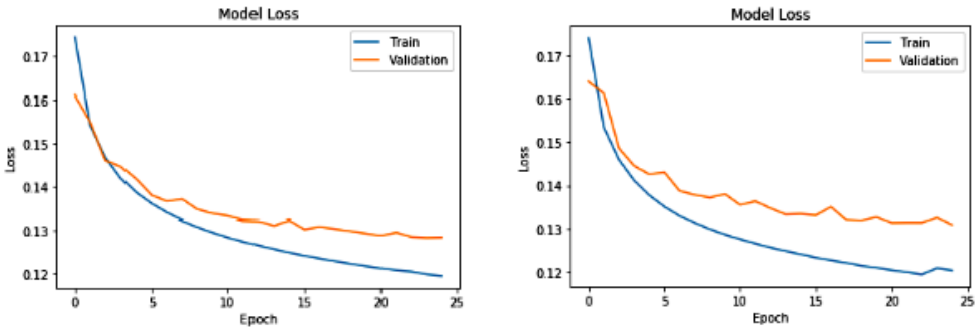
As illustrated in Table 2, the proposed approach ASLSTM outperforms in English all the compared methods on all criteria by successfully returning a significant number of similar questions among the retrieved ones. This good performance indicates that the use of Siamese LSTM along with attention mechanism Manhattan similarity is effective for the QR task. Word embeddings allow to obtain an efficient input representation for LSTM, capturing syntactic and semantic information in a word level. Interestingly, ASLSTM does not require an extensive feature generation owing to the use of a pre-trained model. The results show that the Siamese based approach performs better than the translation and knowledge based methods, which provides evidence that the question representations made by the Siamese LSTM sub-networks can learn the semantic relatedness between pairs of questions and then are more adequate for representing questions in the question similarity task. The Siamese network was trained using backpropagation-through-time under the MSE loss function which compels the LSTM sub-networks to detect textual semantic difference during training. A key virtue of LSTM is that it can accept variable length sequences and map them into fixed length vector representations which can over- come the length and structure's problems in cQA.

Another significant finding is the effectiveness of the attention mechanism which was able to improve the performance of the approach by getting better contextual meaning of the questions. We assume that the attention mechanism managed to boost the similarity learning process by assigning a weight to each element of the question. This weights will then allow to compute which element in the sequence the neural network should more attend.

WEKOS averages the weighted embeddings, which is one of the most simple and widely used techniques to derive sequence embedding but it leads to losing the word order, while in our approach, the LSTMs update their state to get the main context meaning of the text sequence in the order of words. The goal of the Siamese architecture is to learn a function which can map a question to an appropriate fixed length vector which is favor for similarity measurement. Interestingly, it offers vector representation for a very short text fragment that should grasp most of the semantic information in that fragment.

To properly assess the MaLSTM model performance on our similarity prediction problem, we plot training data vs validation data accuracy and loss using the Matplotlib library. The training history of the model is often used to diagnose its behavior and to check whether it is a good fit for the data or could perform better with a different configuration. The loss is computed on training and validation and it shows how well the model is doing for these two sets. It denotes the sum of the errors made for each instance in training or validation sets. Loss is often used to determine the best parameter values for the given model. Obviously, the lower the loss, the better a model is. Once we find the optimized parameters, the accuracy allows to evaluate how accurate our model's prediction is com- pared to the true data. The training accuracy shows how much the model learns to map the input and output, while the validation accuracy tells about its

generalizing ability. For instance, a decreasing validation accuracy means low generalization over the training data.



(a) Results on the English dataset

(b) Results on the Arabic dataset

Figure 8: Epochs vs Loss of Siamese LSTM on the English and Arabic dataset

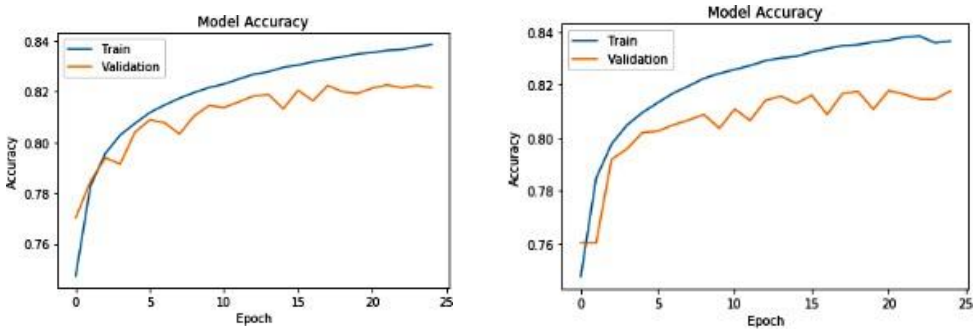
From Figures 8a and 8b which depict the training and validation set loss against the number of epochs[11], we can see that for both English and Arabic there is noconsiderable difference between the training and validation loss. The training loss keeps decreasing after every epoch which means that the model is learning to recognize the specific patterns. Similarly, the validation loss continues to decrease reaching 0.132 and 0.129 for English and Arabic respectively thus, our model is generalizing well on the validation sets. We can say that we have a good fit since both the train and validation loss decreased and leveled off around the same points.

From the plots of accuracy given in Figures 9a and 9b, we observe that we get about 82% and 81% accuracy rate on the validation data for English and Arabic respectively. The model has comparable consistent accuracy on both train and validation sets. Both training and validation accuracy continue to increase without a sudden decrease of the validation accuracy, indicating a good fit. Therefore, we can admit that, whilst the performance on the training set is slightly better than that of the validation set in terms of both loss and accuracy, the model has converged to a stable value without any typical overfitting signs such as the continuous improvement of the training performance, while validation performance worsens.

It is worth mentioning that the accuracy used in the epochs-accuracy plots, is the binary accuracy calculated by Keras, and it implies that the threshold is set at 0.5 so, everything above 0.5 will be considered as correct.

[11] Epoch is when the full dataset is passed both forward and backward through the neural network only once. It usually contains a few iterations

We utilized the simple Manhattan distance which forces the LSTM model to entirely detect the semantic differences during training. In practice, our results are fairly stable across different similarity functions, namely cosine and Euclidean distances. We found that the Manhattan distance has outperformed them on both the English and Arabic datasets as depicted in Tables 3a and 3b which demonstrates that it is the most relevant measure for the case of high dimensional text data.



(a) Results on the English dataset

(b) Results on the Arabic dataset

Figure 9: Epochs vs Accuracy of Siamese LSTM on the English and Arabic dataset

Table 3: Comparison between similarity measures

(a) Results on the English dataset

	P@5	Recall
Manhattan	**0.5033**	**0.5477**
Cosine	**0.3893**	**0.4345**
Euclidean	**0.3393**	**0.3843**

(b) Results on the Arabic dataset

	P@5	Recall
Manhattan	**0.3702**	**0.4146**
Cosine	**0.2562**	**0.3006**
Euclidean	**0.2062**	**0.2506**

The cosine distance has outperformed the Euclidean distance which proves that it is better at catching the semantic of the questions, considering that the direction of the text points can be thought as its meaning, texts with similar meanings will have similar cosine score. Another reason is that cosine distance is calculated using the dot product and magnitude of each vector. Thus, it is only affected by the words the two vectors have in common, whereas the Euclidean measure has a term for every dimension which is non-zero in either vector. We can therefore say that the cosine distance has meaningful semantics for ranking texts, based on mutual term frequency, whereas Euclidean distance does not.

Moreover, we remarked that ASLSTM could find the context mapping between certain expressions mostly used in the same context such as bug and error

message or also need help and suggestions. In addition, ASLSTM was able to retrieve similar questions containing certain common misspelled terms like receive instead of receive, but it failed to capture other less common spelling mistakes like relyable or realible instead of reliable. Such cases show that the proposed approach can address some lexical disagreement problems. Furthermore, there are few cases where ASLSTM fails to detect semantic equivalence, including queries having only one similar question and most words of this latter do not appear in a similar context with those of the query.

Table 4: Question retrieval performance of ASLSTM in Arabic

	WEKOS	**ASLSTM**
P@5	0.3444	**0.3702**
P@10	0.2412	**0.2872**
MAP	0.4144	**0.4540**
Recall	0.3828	**0.4146**

Table 4 shows that ASLSTM outperforms in Arabic the best compared system which gives evidence that it can also perform well with languages having some specificities.

Nevertheless, a major limitation of the presented pipeline is that it ignores the morphological structure of Arabic words. As a matter of fact, the Arabic language is a morphologically-rich and complex language which expresses multiple levels of information at the word level. The variation in character forms appearing in handwritten Arabic has a notable impact on the generation of word embeddings. Therefore, harnessing the word internal structure is crucial to capture semantically similar words. Accordingly, endowing word embeddings with grammatical information such as, the person, gender, number and tense could help to obtain more meaningful embeddings that detect morphological and semantic similarity. In terms of recall, ASLSTM reaches 0.4136 for Arabic which implies that the number of omitted similar questions is not big. As shown in table 5a, Siamese CNN achieves a MAP of 49% and performs better than the state-of- the-art methods, which indicates that it is capable of capturing more informative features, such as the salient words or the sequential structures of questions.

Table 5: Question retrieval performance of Siamese LSTM and CNN

(a) Results on the English dataset

	Siamese LSTM	**Siamese CNN**
P@5	0.5023	0.4943
P@10	0.4188	0.4107
MAP	0.5739	0.5658
Recall	0.5389	0.5288

(b) Results on the Arabic dataset

	Siamese LSTM	**Siamese CNN**
P@5	0.3692	0.3292
P@10	0.2854	0.2454
MAP	0.4513	0.4113
Recall	0.4136	0.4136

CNN could extract syntactic and semantic information from both local

semantic pat- terns and hierarchical structures of the questions. Both CNN and LSTM models are able to preserve the semantics of question words and extract the hidden features from the textual content by means of multiple hidden layers. Siamese LSTM (without the attention layer) performs slightly better than Siamese CNN on both precision and recall. One possible rea- son is that information at question -level is more important than information at word-level for the question similarity task. From table 5b, we can see that the performance on the Arabic dataset are slightly worse than English. The reason behind this could be that there are some errors in Arabic word segmentation.

Table 6: Comparison between similarity measures on Siamese LSTM and CNN

(a) Results on the English dataset

	Siamese LSTM		Siamese CNN	
	P@5	**Recall**	**P@5**	**Recall**
Manhattan	**0.5033**	**0.5477**	**0.4943**	**0.5288**
Cosine	**0.3893**	**0.4345**	**0.3802**	**0.4156**
Euclidean	**0.3393**	**0.3843**	**0.3302**	**0.3656**

(b) Results on the Arabic dataset

	Siamese LSTM		Siamese CNN	
	P@5	**Recall**	**P@5**	**Recall**
Manhattan	**0.3702**	**0.4146**	**0.3292**	**0.4136**
Cosine	**0.2562**	**0.3006**	**0.2151**	**0.2997**
Euclidean	**0.2062**	**0.2506**	**0.1651**	**0.2497**

As shown in the table 6a and 6b, the performance of the Manhattan similarity metric is better than the cosine and Euclidean similarities in terms of precision and recall. Therefore, we adopted the Manhattan distance as it allowed to achieve faster convergence and it is the most relevant one for the dataset we used.

6. Conclusion

In this paper, we proposed an attentive Siamese LSTM-based approach for English and Arabic question similarity learning. We also explored the use of Siamese CNN for the same task. Experiments conducted on large scale Yahoo!Answers datasets have shown that Siamese LSTM outperforms Siamese CNN and some competitive state-of-the art methods. Interestingly, we verified that the proposed approach is capable of modeling complex semantics and covering the context information of question pairs. The use of word embedding helped to fully capture the semantic information at word-level, and obtain the contextual word vector representations of each question pair. An attention mechanism was integrated to let the model give different attention to different words while modeling the questions. In the future, we look forward to building a hybrid deep learning Siamese architecture combining CNN as a feature extractor and LSTM as a classifier to improve the question retrieval performance. In addition, we will conduct experiments on Siamese Transformer using BERT to generate word embeddings.

References

[1] A. Barrón-Cedeno, G. Da San Martino, S. Romeo, and A. Moschitti. Selecting sentences versus selecting tree constituents for automatic question ranking. In *Proceedings of COLING, the 26th International Conference on Computational Linguistics*, pages 2515–2525, 2016.

[2] L. Cai, G. Zhou, K. Liu, and J. Zhao. Learning the latent topics for question retrieval in community qa. In Proceedings of 5th International Joint Conference on Natural Language Processing, pages 273-281, 2011.

[3] X. Cao, G. Cong, B. Cui, and C. S. Jensen. A generalized framework of exploring category information for question retrieval in community question answer archives. In Proceedings of the 19th international conference on WWW, pages 201–210. ACM, 2010.

[4] X. Cao, G. Cong, B. Cui, C. S. Jensen, and C. Zhang. The use of categorization information in language models for question retrieval. In Proceedings of the 18th ACM conference on Information and knowledge management, pages 265–274. ACM, 2009.

[5] W. Chen, C. K. Yeo, C. T. Lau, and B. S. Lee. Leveraging social media news to predict stock index movement using rnn-boost. Data & Knowledge Engineering, 118:14–24, 2018.

[6] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. Attention-based models for speech recognition. In Advances in neural information processing systems, pages 577–585, 2015.

[7] C. Dos Santos, L. Barbosa, D. Bogdanova, and B. Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In Proceedings of ACL and the 7th International Joint Conference on NLP, volume 2, pages 694–699, 2015.

[8] H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu. Searching questions by identifying question topic and question focus. In ACL, volume 8, pages 156–164, 2008.

[9] S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty,

Fuzziness and Knowledge-Based Systems, 6(02):107–116, 1998.

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

[11] C. Johnpaul, M. V. Prasad, S. Nickolas, and G. Gangadharan. General representational automata using deep neural networks. Data & Knowledge Engineering, 122:159–180, 2019.

[12] A. Kamineni, M. Shrivastava, H. Yenala, and M. Chinn akotla. Siamese lstm with convolutional similarity for similar question retrieval. In 2018 International Joint Symposium on Artificial Intelligence and NLP (iSAI-NLP), pages 1–7. IEEE, 2019.

[13] Y. Kim. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.

[14]D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. ICLR, 2014.

[15] R. Malhas, M. Torki, and T. Elsayed. Quir at semeval 2016 task 3: Learning to rank on arabic community question answering forums with word embedding. In Proceedings of SemEval, pages 866–871, 2016.

[16] M. Mohtarami, Y. Belinkov, W.-N. Hsu, Y. Zhang, T. Lei, K. Bar, S. Cyphers, and J. Glass. SLS at semeval- 2016 task 3: Neural-based approaches for ranking in community question answering. In Proceedings of SemEval, pages 828–835, 2016.

[17] J. Mueller and A. Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[18] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10), pages 807–814, 2010.

[19] N. Othman, R. Faiz, and K. Smaïli. Enhancing question retrieval in community question answering using word embeddings. In proceedings of the 23rd International Conference on Knowledge-Based and

Intelligent Information Engineering Systems (KES), 2019.

[20] N. Othman, R. Faiz, and K. Smaïli. Enhancing question retrieval in community question answering using word embeddings. In proceedings of the 23rd International Conference on Knowledge-Based and Intelligent Information Engineering Systems (KES), 2019.

[21] N. Othman, R. Faiz, and K. Smaïli. Manhattan siamese lstm for question retrieval in community question answering. In OTM Confederated International Conferences “On the Move to Meaningful Internet Systems”, volume 11877. Springer, 2019.

[22] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In International conference on machine learning, pages 1310–1318, 2013.

[23] X. Qiu, L. Tian, and X. Huang. Latent semantic tensor indexing for community-based question answering. In ACL (2), pages 434–439, 2013.

[24] S. Romeo, G. Da San Martino, A. Barron-Cedeno, A. Moschitti, Y. Belinkov, W.-N. Hsu, Y. Zhang, M. Mohtarami, and J. Glass. Neural attention for learning to rank questions in community question answering. In Proceedings of COLING, pages 1734–1745, 2016.

[25] S. Romeo, G. Da San Martino, Y. Belinkov, A. Barron-Cedeno, M. Eldesouki, K. Darwish, H. Mubarak, J. Glass, and A. Moschitti. Language processing and learning models for community question answering in arabic. IPM, 2017.

[26] A. Singh. Entity based q&a retrieval. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 1266–1277. ACL, 2012.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1):1929–1958, 2014.

[28] K. Wang, Z. Ming, and T.-S. Chua. A syntactic tree matching approach

to finding similar questions in community-based qa services. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, pages 187–194. ACM, 2009.

[29] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning, pages 2048–2057, 2015.

[30] X. Xue, J. Jeon, and W. B. Croft. Retrieval models for question and answer archives. In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pages 475–482. ACM, 2008.

[31] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies, pages 1480–1489, 2016.

[32] B. Ye, G. Feng, A. Cui, and M. Li. Learning question similarity with recurrent neural networks. In 2017, *IEEE International Conference on Big Knowledge (ICBK)*, pages 111–118. IEEE, 2017.

[33] M. D. Zeiler. Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701, 2012.

[34] K. Zhang, W. Wu, H. Wu, Z. Li, and M. Zhou. Question retrieval with high quality answers in community question answering. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pages 371–380. ACM, 2014.

[35] W.-N. Zhang, Z.-Y. Ming, Y. Zhang, T. Liu, and T.-S. Chua. Capturing the semantics of key phrases using multiple languages for question retrieval. IEEE Transactions on Knowledge and Data Engineering, 28(4):888–900, 2016.

[36] G. Zhou, L. Cai, J. Zhao, and K. Liu. Phrase-based translation model for question retrieval in community question answer archives. In Proceedings of the 49th Annual Meeting of the ACL: Human Language

Technologies-Volume 1, pages 653–662. ACL, 2011.

[37] G. Zhou, Y. Chen, D. Zeng, and J. Zhao. Towards faster and better retrieval models for question search. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management, pages 2139–2148. ACM, 2013.

[38] G. Zhou, T. He, J. Zhao, and P. Hu. Learning continuous word embedding with metadata for question retrieval in community question answering. In Proceedings of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, pages 250–259, 2015.

[39] G. Zhou, Y. Liu, F. Liu, D. Zeng, and J. Zhao. Improving question retrieval in community question answer- ing using world knowledge. In IJCAI, volume 13, pages 2239–2245, 2013.

[40] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 207–212, 2016