COMP61332 Text Mining Coursework 1: Question Classification

Groups 16 & 26

Darsh Jadhav *10806664*, **Huajie He** *10689523*

Jonathan Coutinho *9931026*, **Nuhu Ibrahim** *10723572*

Xinmiao Tang *10618359*, **Yuqi Zhang** *10613157*

Abstract

Question classification plays an essential role in question answering. This paper describes the methodology behind our work and showcases the results obtained from utilising two primary Machine learning (ML) approaches: Bag of Words (BoW) and Bidirectional Long Term Short Term Memory (BiLSTM). Following this, an interpretation and analysis of results is conducted to highlight the significance of our research. Additionally, an ensemble model combining the best aspects of our previous *6* models is implemented to demonstrate future prospects. Experimental results achieved are quite promising, particularly in regards to BiLSTM. Ensemble learning also proved to be an area that requires further exploration.

1 Introduction

Question answering (QA) is an important field within Natural Language Processing (NLP), being the focus of a large amount of current research. QA systems, unlike traditional information retrieval systems, are designed to deliver answers specific to the question asked instead of a spew of information from text(s) (Hermjakob, 2001). For instance, when asked the question "Where was the largest pancake created?", what a user expects is the answer "Rochdale", instead of surveying through documents consisting of key words "largest", "pancake", "created", etc.

A major stage in QA involves classifying the question to the anticipated *type* of the answer, which is done to understand what exactly the question is asking for (Huang et al., 2008). Question Classification (QC) achieves this by classifying the question to a category from a set of N predefined classes.

The Problem: Question Classification

Question Classifiers have been shown to significantly improve the overall performance of a QA system, as mentioned by Loni (Loni, 2011).
The key advantages of its incorporation consist of:

- Reduction in overall search space and greater probability of selecting the *true* answer; and
- Better informed search strategy/query to search for and determine possible answers.

Our research demonstrates and evaluates the performance of automated question classifiers *training* on the "learning question classification" Training set 5 data (5500 questions) used by *Li et al.* (Li and Roth, 2002), and the (Text Retrieval Conference) *TREC 10* data set (500 questions) as a *test* set.

The classifiers utilise two approaches: Bag-of-words (BoW) and Bidirectional Long Term Short-Term Memory (BiLSTM). Furthermore, before classification is performed, words from the training data sets are embedded (assigned values and stored as vectors) using two word embedding methods: *Randomly initialised* embedding, *Pre-trained* embedding.

This paper focuses on the effects of these two main approaches while utilising different preprocessing methods. The overall aim is to build a classifier that is able to accurately classify questions according to one of *50* predefined classes (Li and Roth, 2002). Using the previous example, the output of the *ideal* classifier when given the question would be "*LOC:other*", as the question is asking for "where" the largest pancake was created (since it is not asking for a "city" or "country", "other" is used). As seen in the example above, there is often a lot of ambiguity in questions ("where" could refer to a building, water body, or even another planet!). Additionally, it is difficult for question classifiers to ascertain the specific context of a question without any previous information (Huang et al., 2008) e.g.: "Who is Tom?". "Tom" here could refer to the name of a human, animal, or even an object.

Our research attempts to solve these issues by testing and experimenting with the BoW and BiLSTM ML techniques, along with an ensemble model that combines their best aspects.

2 Classifiers: Background

This section will introduce the concepts behind each of the two classifiers and then highlight their significance and hint at some of the future possibilities they offer.

The experiments we conduct involve the use of Pre-trained embedding with *Freeze* or *Fine-tuning*, and Randomly initialised embedding with *Freeze*. As a result, *6* different models will be provided (3 types of word embeddings for the two ML approaches). In addition, we also propose an Ensemble model. In this experiment, we will use *F-score* (Sasaki, 2007) to evaluate our model and compare results. Confusion matrices and Error plots are also provided to visualise our findings and illustrate the pros and cons of each model.

2.1 Bag-of-words

A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modelling, such as with machine learning algorithms. The approach is simple and flexible, and can be used in a myriad of ways for extracting features from documents (Yoav and Graeme, 2017). The theoretical basis of this model is that two articles are similar when they have similar content. This model constructs word vectors by counting the frequency of words appearing in different documents. As a result, the order and structure of words in the document are irrelevant.

2.2 BiLSTM

Long Term Short Term Memory, or LSTM (Hochreiter & Schmidhuber, 1997) is a popular variant of Recurrent Neural Networks (RNN). It can use the context information from previous tokens to alleviate the gradient vanish problem of RNN. BiLSTM makes some improvements based on LSTM, it utilises both the previous and future context by processing the sequence on two directions, and generate two independent sequences of LSTM output vectors. One processes the input sequence in the forward direction, while the other processes the input in the reverse direction. The output at each time step is the concatenation of the two output vectors from both directions (Tan et al., 2015).

2.3 Feed-forward neural network

A feed-forward neural network (feed-forward neural net) is an instance of supervised machine learning. Typically, it has three types of nodes: *input* nodes, *hidden* nodes, and *output* nodes. A key feature of a feed-forward neural net is that the units are connected with no cycles. The outputs from the units in each layer are passed on to units in the next (higher) layer. Another advantage is that feed-forward neural nets are fully-connected. This means that each unit in the current layer uses the outputs from all the units in the previous layer as input. Therefore, there is a link between every pair of units from two adjacent layers (Jurafsky, 2000). Additionally, feed-forward neural nets can have one or multiple hidden layers.

2.4 Ensemble learning and Future work

Ensemble learning involves training multiple models to achieve the same task. Many methods of how to go about this exist, but the most popular are Bagging and Boosting.

As previously mentioned, question classification is an important pre-step of question answering. Our research utilises ensemble learning to improve our classifier's accuracy because it will greatly influence the overall performance of a question answering system. Future work would entail a thorough exploration of results obtained from different ensemble models and different classification strategies such as Gated Recurrent Units (*GRUs*) (Chung et al., 2014) or Support Vector Machines *SVMs* (Zhang and Lee, 2003), for example.

3 Methodology

3.1 Data Pre-processing

In this experiment, we used five different data processing approaches. The first is to convert all uppercase letters to lowercase. The purpose is to ensure the consistency of words and prevent the inconsistency of word vectors due to *capitalization*. For instance, "For example", "for example" should be consistent. The second is to remove some *stop words* using a *dictionary*. The specific method is to iterate through all the sentences and remove words that correspond to those in the dictionary. The third is to remove special punctuation marks e.g.: [,.?!']. The fourth is to replace abbreviations e.g.: "*it's*" replaced by "*it is*". By expanding abbreviations, we ensure that the referential relationship between sentences is clear. The fifth is to remove blanks.

After the input text is pre-processed, a large number of continuous blanks and line breaks will appear, which are uniformly replaced with a single blank here. The purpose is to facilitate the reading and processing of data.

3.2 Word Embeddings

3.2.1 Randomly initialized word embeddings

In this part, we need to build two dictionaries. The first dictionary is to count the number of occurrences of all words that appear in the data text. The second dictionary is to count the number of occurrences of classification labels. The specific method is to construct a set(), use enumeration to get each word and the number of their occurrence. The same applies to the statistical methods of classification labels.

3.2.2 Pre-trained word embeddings - Glove

In this project, we choose *Glove* as our pre-trained word embedding. To load the word representation efficiently, we ignore the words which do not appear in the vocabulary. After that, we get the corresponding vector from Glove for every word in the dictionary and convert questions to vectors represented by their pre-trained word embeddings.

3.3 Bag of Words (*BoW*)

The first step of implementing the Bag-of-Words model is to split the sentences into unique words as the model *vocabulary*. After that, for each question, we need to score the words according to the vocabulary. This aims to turn each question into a vector that can be used as input for a deep learning model. In this case, there are more than 2000 unique words in the vocabulary, so we create a binary vector where its length equals the length of vocabulary for each question. We mark the presence of words as a boolean value, 0 for *absent*, 1 for *present*. For new questions that contain words outside of the vocabulary, we regard these words as "*unknown*" and ignore them.

3.4 Bidirectional Long Term Short-Term Memory (*BiLSTM*)

Here we use the LSTM model provided by Pytorch. We needed to make some changes to the model since BiLSTM is required in this experiment. The main changes are as follows:

1) Support parameter "bidirectional" to True;
2) Multiply the number of layers by 2 when mapping the hidden layer to the tag space; and
3) Combine the results of BiLSTM in two different directions.

During specific training, each sentence is inputted in turn and the prediction result is obtained, and then the backward propagation calculation of the gradient is completed.

3.5 Error-handling

In order to improve the Robustness of the program, we carried out error handling. Details are as follows:

1) Check the content of the configuration file and throw exceptions when invalid configuration combinations are entered;
2) Perform read and write file verification and throw exceptions when files cannot be read or written;
3) Verify all file structures and throw exceptions when they do not meet required form;
4) Check the execution command written from the command line and throw exceptions when it does not match expected format; and
5) Check all parameters and throw exceptions when the type is abnormal or there is value overflow or underflow.

Whenever any of the above exceptions occur, the error handling system immediately interrupts the process and reports the detailed error information.

4 Experimentation: Results and Analysis

The following section goes into detail about the experimental results and how they were obtained. Below are tables consisting of the metrics we used to assess the performance of the classifiers, followed by an analysis in the latter paragraphs, and an ablation study.

Pre-trained Word Embedding From the results, it is evident that when *pre-trained embedding* is used, the BoW classifier demonstrates similar performances with only slight differences in the *Accuracy, F-Score, Precision and Sensitivity* for both the *Freeze* and *Fine-tune* options. On the contrary, the BiLSTM classifier showcases a more significant increase in all 4 metrics when Fine-tuning is used.

Figure 1: Results from the **BoW** classifiers with the two types of Word Embedding

	Pre-trained Word Embeddings				Random Word Embeddings	
	Freeze		Fine-tune		Freeze	
BoW	Train	Test	Train	Test	Train	Test
Accuracy (%)	57.1	57.4	93.7	72.2	79.6	72.8
F-Score	0.3266	0.2935	0.9313	0.5267	0.6341	0.5295
Precision	0.3181	0.3350	0.9216	0.5714	0.5755	0.5401
Sensitivity	0.3408	0.8044	0.9424	0.6496	0.8729	0.8054
Params Description	lowercase:True remove_stop_words:False remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 learning_rate:0.001 freeze:True		lowercase:True remove_stop_words:True remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 learning_rate:0.014 freeze:False		lowercase:True remove_stop_words:True remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:random epoch:10 embedding_dim:32 learning_rate:0.001 freeze:None	

Figure 2: Results from the **BiLSTM** classifiers with the two types of Word Embedding

	Pre-trained Word Embeddings				Random Word Embeddings	
	Freeze		Fine-tune		Freeze	
BiLSTM	Train	Test	Train	Test	Train	Test
Accuracy (%)	93.1	73.4	96.5	75.4	95.4	72.4
F-Score	0.8917	0.5254	0.9476	0.5672	0.9245	0.5035
Precision	0.8629	0.5927	0.9350	0.6104	0.9024	0.5858
Sensitivity	0.9307	0.6185	0.9636	0.6662	0.9548	0.6162
Params Description	lowercase:True remove_stop_words:True remove_special_characters: False replace_abbreviations: False remove_white_space: False embedding:pretrained epoch:10 embedding_dim:300 hidden_dim:32		lowercase:True remove_stop_words:True remove_special_characters:False replace_abbreviations:False remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 hidden_dim:32 learning_rate:0.06		lowercase:True remove_stop_words:False remove_special_characters:False replace_abbreviations:False remove_white_space:False embedding:random epoch:10 embedding_dim:64 hidden_dim:32 learning_rate:0.06 freeze:None	

Random Word Embedding In context to the BoW classifier, it is quite interesting to note that although it has low values in 3 out of the 4 metrics, its *Sensitivity* is high compared to when *pre-trained embedding was used*. This indicates that it classifies true positives well. BiLSTM demonstrates significantly worse performance compared to Pre-trained word embedding, with lower values in all 4 metrics.

After running our Ensemble model implementation, we achieved the following on the overall Test set (with the 5 models having overall average of 58% accuracy on the development set):

Ensemble Model			
Accuracy (%)	F-Score	Precision	Sensitivity
72.6	0.7009	0.71998	0.726

Additional Tests We conducted experiments where we altered some of the parameters and settings and made the following observations about our best performing model, the BiLSTM pre-trained embedding (fine-tune) classifier: It performed better when *Stop words* were not removed, with a 3% increase in its F-score, precision and accuracy. It has the worst classification rate for the classes: *diseases and medicine*, *products* and the *title of a person*. On the contrary, it had the best classification rate for: *weight*, *speed* and *the lasting time of something*.

Interpretation Considering the results obtained, BiLSTM with Pre-trained word embedding (*Fine-tune*) is the best question classifier, obtaining the highest values in all 4 metrics in regards to the Test set. Overall, it is clear that BiLSTM performs significantly better than BoW. Although the ensemble model did not have the best performance, it will be interesting to experiment with a combination of different classifiers (other than BoW & BiLSTM) and perhaps utilising another ensemble approach altogether. If there were no time constraints, this would be the direction our future work would follow.

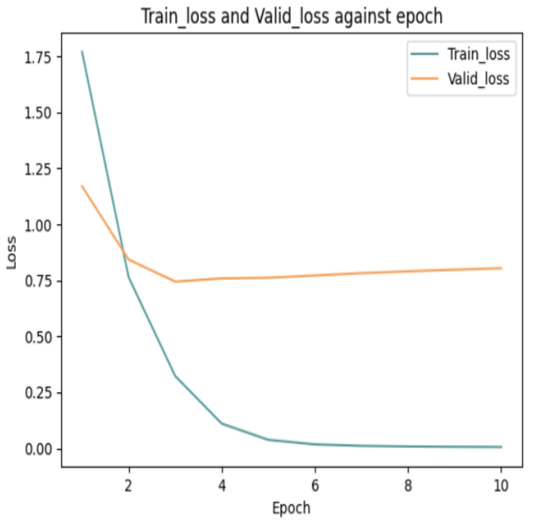
5 Visualisation of Results

On the following page is a plot of the *Training and Validation loss* versus *Epochs*, when training our best model, *BiLSTM with pre-trained embedding (fine-tune)*, to further cement our findings.
Additionally, the *Confusion matrix* for the classifier was too big to fit into this paper but it was used to obtain all the Evaluation metrics used in the *Experimentation: Results and Analysis* and can be obtained using the source code.

6 Conclusion

The main objective of this research was to explore the differences between BoW and BiLSTM, and then build an accurate question classifier. This was completed after performing a survey of the work done in these areas and developing models

Figure 3: Train. and Valid. Loss over Epochs plot



accordingly. Our results indicated that BiLSTM with pre-trained embedding (fine-tune) was the best performing classifier for Question Classification. Ensemble learning was also attempted to combine the two approaches; we believe that future work needs to be done in this area particularly with a wider variety of models to incorporate their best aspects.

7 Group Statement

Tackling this coursework as a group has helped us learn from one another and developed our teamwork capabilities. The need to work together remotely has also motivated us to use many different state-of-the-art remote coding and working tools that will be useful in the professional scene. Writing a short paper was also quite enjoyable and allowed us to experience authoring a research paper as a group, something a few of us had not partaken in before.

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Ulf Hermjakob. 2001. Parsing and question classification for question answering. In *Proceedings of the ACL 2001 workshop on open-domain question answering*.

Zhiheng Huang, Marcus Thint, and Zengchang Qin. 2008. Question classification using head words and their hypernyms. In *Proceedings of the 2008 Conference on empirical methods in natural language processing*, pages 927–936.

Dan Jurafsky. 2000. *Speech & language processing*. Pearson Education India.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Babak Loni. 2011. A survey of state-of-the-art methods on question classification.

Y Sasaki. 2007. The truth of the f-measure. university of manchester. Technical report, Technical Report, Version. Available online: http://www. flowdx. com/F ....

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.

Goldberg Yoav and H Graeme. 2017. Neural network methods in natural language processing. *Morgan & Claypool: San Rafael, SR, USA*, pages 104–113.

Dell Zhang and Wee Sun Lee. 2003. Question classification using support vector machines. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32.

8 Appendices

8.1 Appendix A

Figure 4: Results from the **BoW** classifiers with the two types of Word Embedding

	Pre-trained Word Embeddings				**Random Word Embeddings**	
	Freeze		Fine-tune		Freeze	
BoW	Train	Test	Train	Test	Train	Test
Accuracy (%)	57.1	57.4	93.7	72.2	79.6	72.8
F-Score	0.3266	0.2935	0.9313	0.5267	0.6341	0.5295
Precision	0.3181	0.3350	0.9216	0.5714	0.5755	0.5401
Sensitivity	0.3408	0.8044	0.9424	0.6496	0.8729	0.8054
Params Description	lowercase:True remove_stop_words:False remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 learning_rate:0.001 freeze:True		lowercase:True remove_stop_words:True remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 learning_rate:0.014 freeze:False		lowercase:True remove_stop_words:True remove_special_characters:True replace_abbreviations:True remove_white_space:False embedding:random epoch:10 embedding_dim:32 learning_rate:0.001 freeze:None	

8.2 Appendix B

Figure 5: Results from the **BiLSTM** classifiers with the two types of Word Embedding

	Pre-trained Word Embeddings				**Random Word Embeddings**	
	Freeze		Fine-tune		Freeze	
BiLSTM	Train	Test	Train	Test	Train	Test
Accuracy (%)	93.1	73.4	96.5	75.4	95.4	72.4
F-Score	0.8917	0.5254	0.9476	0.5672	0.9245	0.5035
Precision	0.8629	0.5927	0.9350	0.6104	0.9024	0.5858
Sensitivity	0.9307	0.6185	0.9636	0.6662	0.9548	0.6162
Params Description	lowercase:True remove_stop_words: True remove_special_characters: False replace_abbreviations: False remove_white_space: False embedding:pretrained epoch:10 embedding_dim:300 hidden_dim:32		lowercase:True remove_stop_words:True remove_special_characters:False replace_abbreviations:False remove_white_space:False embedding:pretrained epoch:10 embedding_dim:300 hidden_dim:32 learning_rate:0.06		lowercase:True remove_stop_words:False remove_special_characters:False replace_abbreviations:False remove_white_space:False embedding:random epoch:10 embedding_dim:64 hidden_dim:32 learning_rate:0.06 freeze:None	