Text Mining CW1: Question Classification

Jie Wang
dept. Computer Science
University of Manchester
e63736jw

Rong Jian Yee
dept. Computer Science
University of Manchester
w74390ry

Hao Cao
dept. Computer Science
University of Manchester
b76967hc

***Abstract*—This paper is intended to be a report for Text Mining module, investigating questions classifier using two methods of sentence representation, Bag-of-Words(BOW) and Bidirectional Long Short Term Memory(BiLSTM) respectively. In this paper, different methods of word embedding is also considered in the comparative analysis. In essence, the experiments show that using BiLSTM as a sentence representation method is better overall.**

***Index Terms*—Bag of Words, BiLSTM, Question Classification, Word embedding**

I. Introduction and Background

A. Problem Identification

The objectives of this paper are to perform a comparative analysis of question classifiers using two methods of sentence embedding, bag-of-words and BiLSTM. Which question classifier will perform better? Question classification essentially tries to classify or label a question into a category that represents the answer type[1]. One example in the dataset that we are using is the question ”What are liver enzymes?”, the classifier will try to label the question into the class ”DESC: def” which means that the answer to this question is of type ”Description: definition”.

B. Importance of Question Classification

With the advancement of technology in the past decades, question classification has become an essential part of the question-answering system. R.Mervin[2] described that a question-answering system is an informational retrieval system which aims to provide a direct answer in response to a query. To build a successful question-answering system, it is necessary to train a good question classifier. Dan et al.[3] concluded that the overall performance of a question-answering system depends on the classification of questions.

It is therefore essential to develop question classifier with high accuracy since question answering system exists in many of our daily technology such as voice assistant in smart devices and chatbots that are embedded in websites.

C. Challenges and Difficulties

Since question classification and question answering are closely related, they share the same difficulties and challenges. Lorena and Elinda[4] in 2017 addressed some of the challenges faced by researchers in this field. Here are a few of the challenges Lorena and Elinda mentioned in their paper[4]:

- **Entity Identification and Linking:** This is the challenge where the system is unable to identify the subject entity correctly in a question and link it to the knowledge base.
- **Lexical gap in questions:** This is when there are words or phrases that are not captured in the knowledge base. This challenges the system correctly classify the question and hence provide inaccurate answers.

The two above challenges quoted are only a small part of research difficulties in the field of question classification and answering. These difficulties are also the main active research field in NLP.

II. Methodology

A. Sentences to Vectors

Since Machine Learning models can only have numerical value as input, converting text values into numerical values is essential. In the experiment, two methods of sentence representation are used.

*1) **Bag-of-words(BOW)**:* After word embedding in the preprocessing steps which will be mentioned in the later section, sentence representation is obtained by doing the calculation shown in the equation below. The equation[5] for sentence representation is given as follows:

$$vec_{bow}(s) = \frac{1}{|bow(s)|} * \sum_{w \in bow(s)} vec(w) \qquad (1)$$

where s is a sentence, and $vec_{bow(s)}$ is the vector representation for s. The vector representation for word w is expressed by $vec(w)$.

*2) **Bidirectional Long Short Term Memory(BiLSTM)**:* In essence, LSTM is an alternative recurrent neural network that was introduced by Sepp Horchreiter and Jurgen Schmidhuber[6] in 1997, where the main difference is LSTM has 4 neural net layers that regulate the cell states for adding and removing information. BiLSTM is simply an extension of LSTM where additional LSTM is added on top that allows input flows in the opposite direction hence bidirectional.

Utilising the word embedding, the BiLSTM is able to learn a vector representation of the sentence denoted as follows[5]:

$$vec_{bilstm}(s) = BiLSTM(s) \qquad (2)$$

where $BiLSTM(s)$ is the vector obtained from BiLSTM.

B. Models

The model used to classify questions is a feed-forward neural network with a softmax output layer. The hyperparameters used in the classifier will be shown in detail in the experimental setup section. The neural net used only consist of an input and output layer without any hidden layer since the dataset used only consists of 5500 samples.

III. Experimental Setup

A. Data

The training and testing data used in this paper is a subset of the dataset obtained from ”Experimental Data for Question Classification”[7] which consists of 5500 samples with 6 coarse classes and 50 fine classes.

The training data is split randomly into a 9:1 ratio where 1 portion of the training data is used for development and hyperparameter tuning. After reading the training data, it is separated into sentences and fine labels.

B. Preprocessing

Before splitting the dataset, the coarse and fine labels are extracted from it to form two label dictionaries with unique numerical values. Then, all sentences in the dataset are converted into lowercase.

After splitting the training data into a training set and a development set, an initial vocabulary list that includes term frequency is created from the sentences in the training set. The vocabulary list then goes through a series of cleaning shown in the preprocessing Algorithm 1 to acquire a clean vocabulary list.

On the other hand, the labels dictionary is extracted from the dataset and then separate into fine label dictionary and coarse label dictionary follow by transforming

Algorithm 1: Preprocessing

```
Input: Vocab List, $K$, stopwords_list
Initialise clean_vocab_list
for $vocab \in Vocab\ List$ do
    if $vocab\ frequency \geq K$ & $vocab \notin stopwords\_list$ then
        vocab = Regex(vocab);
        clean_vocab_list ← vocab;
    end
end
Output: clean_vocab_list
```

An additional vocabulary #pad# is added at the beginning of the clean vocabulary list, for padding so that all sentences are of the same length.

After obtaining a clean vocabulary list, word embedding is created using two methods:

- **Random:** Random embedding is achieved by using the utility function ”nn.embedding” from PyTorch with the ”padding_idx” set to 0 since the #pad# vocab is the 0^{th} element in the vocabulary list.
- **Pretrained:** The pretrained word embedding is obtained from ”global vector(gloVe)”[8] and then extracts the word vector that matches the vocabulary list from the previous step.

Once the word embeddings are created, they will be sent to the two sentence representation methods mentioned in the methodology to convert sentences into vectors.

C. Hyperparameters

In this section, the hyperparameters used in all the models are presented in Table I and Table II for Coarse Classes and Fine Classes respectively. These hyperparameters are used to produce the best results.

TABLE I
Hyperparameters for Coarse Classes

Name	BOW		BiLSTM	
Learning Rate	0.005	0.005	0.0037	0.0037
Max. Length	36	36	36	36
Hidden dim.	300	300	300	300
Batch Size	400	400	400	400
Embedding dim.	800	300	800	300
Word Embedding	Random	Pre-trained	Random	Pre-trained

TABLE II
Hyperparameters for Fine Classes

Name	BOW		BiLSTM	
Learning Rate	0.005	0.005	0.0037	0.0033
Max. Length	36	36	36	36
Hidden dim.	600	300	600	600
Batch Size	400	400	400	400
Embedding dim.	2000	300	800	300
Word Embedding	Random	Pre-trained	Random	Pre-trained

D. Performance Indicator and Evaluation Metrics

To evaluate the performance of the classifiers, accuracy and Macro F1 score are used and utilize the confusion matrix to observe the final classification result.

IV. Results and discussion

Table III and Table IV illustrated the results obtained from the experiments of Coarse Classes and Fine Classes respectively.

Table III shows that all models using BiLSTM sentence representation outperform the models using BOW with maximum Maro F1-score of 85.54% when Random word embedding is used. However, it should be noted that when running the models on test dataset, models that used BOW sentence representation performs significantly better, this might due to the high similarity in context information of training and development set, however test set contain very different context information therefore the model show signs of overfitting.

In Table IV, it is evident that the results for classifying Fine Classes are all worst than classifying Coarse Classes. This is due to the high number of classes compared to Coarse Classes.

TABLE III
Experimental Results for Coarse Classes

No.	Word Embedding		Sentence Rep.	Macro F1
1	Random	-	BOW	Dev:74.61% Test:80.96%
2	Pre-trained	Freeze	BOW	Dev:69.60% Test:77.57%
3	Pre-trained	Fine-tune	BOW	Dev:75.58% Test:61.14%
4	Random	-	BiLSTM	Dev:85.54% Test:34.03%
5	Pre-trained	Freeze	BiLSTM	Dev:83.12% Test:30.05%
6	Pre-trained	Fine-tune	BiLSTM	Dev:84.98% Test:21.69

TABLE IV
Experimental Results for Fine Classes

No.	Word Embedding		Sentence Rep.	Macro F1
1	Random	-	BOW	Dev:64.03% Test:62.79%
2	Pre-trained	Freeze	BOW	Dev:55.05% Test:18.02%
3	Pre-trained	Fine-tune	BOW	Dev:54.90% Test:57.09%
4	Random	-	BiLSTM	Dev:67.82% Test:15.93%
5	Pre-trained	Freeze	BiLSTM	Dev:58.31% Test:0.5%
6	Pre-trained	Fine-tune	BiLSTM	Dev:56.72% Test:3.34%

In addition, classes imbalance is also a notable limitation that contributes to the poor performance for classifying Fine Classes. The number of sample for each unique Fine Classes is shown in Table VII in the Appendix. Overall, model with BiLSTM sentence representation also outperforms BOW with the highest Macro F1-score of 67.82%. In addition, the same observation is made where the results on test dataset are far worst than development dataset especially when freezing pre-trained word embedding.

In addition, Table V and Table VI is two example confusion matrix for Coarse Classes. It is obvious that the classifier is having trouble classifying the ”ENTY” class. Where most of the false positive classified in ”HUM” class, this mean that these two classes are closely related.

On the other hand, due to the extreme imbalance Fine Classes, therefore the it is very difficult for the classifier to learn to classify the minor classes. The potential redemption for this issue is to perform data augmentation.

To sum up, classifying Fine classes is much more challenging than Coarse Classes is because of limited amount of data which leads to extreme imbalance data.

A. *Ablation*

From the results obtained from the experiments, fine-tuning the word embedding will improve the overall results when classifying Coarse Classes, and the effect is significant when BOW representation is used. However, this observation is not found on classifying Fine Classes.

Initially, we expect using pre-trained word embedding will outperform random initialised word embedding, but this is not the case for results obtained. The experiments results show that random word embedding performs approximately 10% better in terms of Macro F1-score. The potential reason for this is when extracting the word embedding from gloVe that matches our vocabulary list, the features of the word embedding do not matches the task question classification.

TABLE V
Confusion Matrix for Coarse Classes Pretrained Embedding

	ABBR	DESC	ENTY	HUM	LOC	NUM
ABBR	2	2	2	0	1	0
DESC	0	106	18	1	2	5
ENTY	0	9	92	4	3	0
HUM	0	2	26	75	2	1
LOC	0	2	10	4	78	1
NUM	0	5	7	0	0	86

TABLE VI
Confusion Matrix for Coarse Classes Random Embedding

	ABBR	DESC	ENTY	HUM	LOC	NUM
ABBR	4	3	0	0	0	0
DESC	0	120	10	0	1	1
ENTY	0	5	87	15	1	0
HUM	0	3	11	89	2	1
LOC	0	2	4	6	82	0
NUM	0	1	3	1	0	93

V. Conclusion

It is safe to conclude that overall, the question classifier that uses BiLSTM for sentence representation outperforms the ones that use BOW sentence representation by a small margin.

A. *Limitation*

It should be noted that due to the small amount of data used in this experiment, the neural network in the classifier only consists of a minimal number of layers. This is because with a small amount of data if more layers are added to the neural network the performance of the model will not improve and might not even converge if too many layers are added.

B. Future Work

In the future, with more manpower and time, a potential extension to this experiment is to utilise a larger amount of data for training in order to use a sophisticated neural network. In addition, it is also possible to perform over or undersampling methods to ensure the models are trained with a balanced dataset. Furthermore, a potential future work is to use Bidirectional Encoder Representations from Transformers(BERT) for classification and Neural Architecture Research(NAS) to explore more suitable neural networks for the problem.

References

[1] S. Jayalakshmi and A. Sheshasaayee, "Question classification: A review of state-of-the-art algorithms and approaches," *Indian J. Sci. Technol*, vol. 8, no. 29, pp. 1–4, 2015.

[2] R. Mervin, "An overview of question answering system," vol. 1, Oct. 2013.

[3] D. Moldovan, M. Pasca, S. Harabagiu, and M. Surdeanu, "Performance issues and error analysis in an open-domain question answering system.," Jan. 2002, pp. 33–40.

[4] L. Kodra and E. Kajo, "Question answering systems: A review on present developments, challenges and trends," *International Journal of Advanced Computer Science and Applications*, vol. 8, Jan. 2017. DOI: 10.14569/IJACSA.2017.080931.

[5] N. N. Riza Batista-Navarro, *Coursework 1: Question classification comp61332 text mining*, Feb. 2023.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[7] D. R. Xin Li, *Experimental data for question classification*, 2002. [Online]. Available: https://cogcomp.seas.upenn.edu/Data/QA/QC/.

[8] C. D. M. Jeffrey Pennington Richard Socher, *Global vectors for word representation.*

Appendix

TABLE VII
Label Count

Label	Count	Label	Count
DESC:manner	277	LOC:mount	22
ENTY:cremat	208	NUM:money	72
ENTY:animal	113	ENTY:product	43
ABBR:exp	71	NUM:period	76
HUM:ind	963	ENTY:substance	42
HUM:gr	190	ENTY:sport	63
HUM:title	26	ENTY:plant	14
DESC:def	422	ENTY:techmeth	39
NUM:date	219	NUM:volsize	14
DESC:reason	192	HUM:desc	48
ENTY:event	57	ENTY:instru	11
LOC:state	67	ABBR:abb	17
DESC:desc	275	NUM:other	53
NUM:count	364	NUM:speed	10
ENTY:other	218	ENTY:word	27
ENTY:letter	10	ENTY:lang	17
LOC:other	465	NUM:perc	28
ENTY:religion	5	NUM:code	10
ENTY:food	104	NUM:dist	35
LOC:country	156	NUM:temp	9
ENTY:color	41	ENTY:symbol	12
ENTY:termeq	94	NUM:ord	7
LOC:city	130	ENTY:veh	28
ENTY:body	17	NUM:weight	12
ENTY:dismed	104	ENTY:currency	5