Duplicate Question Detection (DQD) using Siamese LSTM

Mamata Shrestha [a], Mohit Dhungana [b], Subin Panta [c], Ujjwal Pudasaini [d] , Santosh Giri [e]

[a, b,c,d]*Department of Electonics and Computer Engineering, Kathford College of Engineering and Management, Lalipur, Nepal*

[e]*Department ofElectonics and Computer Engineering,Pulchowk Campus, IOE*

Corresponding Email : [a] mamatashrestha36@gmail.com, [b] mohitdhungana@yahoo.com,[c]subinpanta99@gmail.com, [d] uzol12345@gmail.com,[e] santoshgiri@ieee.org

***Abstract*—Natural Language processing (NLP) is a subdivision of artificial intelligence. It allows computer to understand, process and generate human languages. Natural Language Processing is used for sentiment analysis, text classification and categorization, language generation, multi-document summarization and so on. In this paper, a classifier model was designed to classify questions as duplicate by detecting their semantic similarity. A classifier model with embedding layer and LSTM layer was designed and the distance between the vector representations of two sentences were calculated using Manhattan distance. Model was trained with the dataset provided by Kaggle for non-commercial purposes. The system has provided 85% final training accuracy. After training, the performance of the system was gauged with the testing dataset. The testing accuracy was found to be 81%.**

Keywords—NLP, Artificial Intelligence, Semantic Similarity, LSTM, Embedding, Manhattan Distance

I. INTRODUCTION

The concept of neural network was first introduced by Walter Pitts and Warren McColloch in 1943 [1]. Neural network research could be split into two approaches based on this model. First approach centered on applying neural networks to AI and the second centered on biological processes within the brain. Henry J. Kelley [2] and Arthur E. Bryson [3] were given credit for the development of backpropagation in the 1960's. Backpropagation worked by fine-tuning the weights to improve the network until desired set of results were obtained [4].Activation function derivatives were to be known prior at the network design time in backpropagation.

The credit for creation of Word2vec was given to team of researchers led by Tomas Mikolov at Google [5]. By taking large corpus texts as inputs, word2vec creates a vector space of several dimensions and assigns corresponding vector to each individual word. The word vectors sharing similar contexts with other words within the corpus are kept closer to one another in vector space [6]. Word2vec consists of two models: continuous bag-of-words (CBOW) and continuous gram, either of which can be utilized to produce distributed representation of words. In [7] it was found that CBOW is faster than skip-gram but gives better results for infrequent words. Word2vec model can be trained with hierarchical soft-max and/or negative sampling.

In 1997 [8] Hochreiter and Schmidhuber introduced Long Short-Term Memory networks (LSTM), a special kind of artificial Recurrent neural network (RNN) which were later refined and popularized by many other people. LSTM model removed one of the key limitations of basic RNN which was vanishing gradient for long sequences.

II. RELATED WORK

In [9] Jaccard coefficient was used to measure the similarity between two questions. Here the dataset used was created with the help of a Chinese question-answer forum known as Baidu Zhidao. The system was trained on 3 million question-answer pairs and tested on 3000 pairs. The system received F-score of 60.29.

Three approaches to DQD were used in [10] : first approach depends upon rule based heuristics, second uses a approach of machine learning classifier which is based on lexical and semantic features, and the third approaches through a deep convoluted neural network. Jaccard Index was used in the first approach. Here, if the Jaccard index exceeded a certain thresh-old then the two questions were considered to be similar. The best valued accuracy achieved on the training data was used to determine the threshold. SVM (Support Vector Machine) was used in the second approach while DCNN (Deep Convolutional Neural Network) was used in the third. In this system it was observed that the best DQD approach was dependent not

only upon the type of the approach used but also relied on the volume of the dataset used. . For small training datasets it was seen that the rule-based heuristics provided better results out of the three while for larger dataset deep convoluted neural network seemed to provide better results. In [11] deep neural network approach was used. . This achieved the best accuracy in the SemEval-2016 Task 1 "QuestionQuestion" subtask. The system obtained an accuracy of 0.73035 in terms of Pearson correlation.

III. METHODOLOGY

A) Data Preprocessing

The dataset made available by Kaggle for non-commercial purposes was used for training and testing the classifier model. This dataset is a collection of 404,351 questions pairs with 255,045 negative samples (non-duplicates) and 149,306 positive samples (duplicates). Each question pair is preprocessed to convert raw text into the list of word indices.

B) Vector Representation

Each word present in sentences are represented by vector coordinates which specifies their position in a vector space. Word2vec has made it possible to obtain word embeddings. These embedding are generated using neural network. The credit for the introduction of Word2Vec concept goes to Tomas Mikolov. Google Word2Vec pre-trained model was used here to obtain the vector representation for all the words in the vocabulary. Each word has a vector of 300 dimensions. We make use of these vectors and word indices to make embed-ding matrix. Embedding matrix is N*300 dimensions matrix of float value where N represents total number of unique words in the vocabulary. The indices value of words provides the row number where its vector is present in embedding matrix.

C) Designing NL Classifier

As show in figure 1 the model expects two inputs of equal lengths. Each input is padded with zeros to convert them into vectors of fixed length. Inputs consists of sequences of word indices where indices are used to identify each word in the sentence. The embedding layer generates embedding matrix for each input by looking up the indices and their corresponding embedding vector. At this layer we have two embedding matrices, one for each input. These embedding matrices are fed into LSTM layer which returns a vector of 50 dimensions for each input.

Manhattan Distance: The vectors returned by LSTM layer is used to measure the semantic similarities. We used Manhattan Distance between these vectors and put them through similarity function. The Manhattan distance between two points is defined as:

$$d(A,B) = \sum_{i=1}^{\infty} |(Ai - Bi)| \qquad (1)$$

Similarity Function: This function returns the value between 0 and 1. The sentences are said to have semantic similarity if this function returns value closer to 1 and vice-versa.

$$s(A,B) = e^{-d(A,B)} \qquad (2)$$

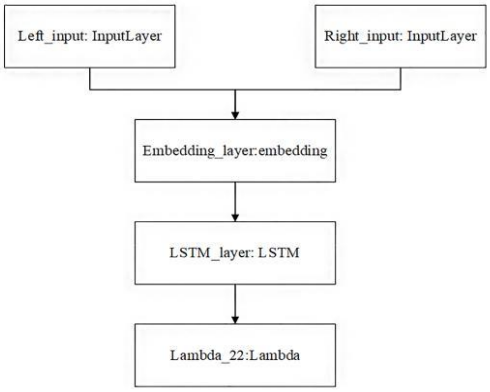

Fig. 1. NL Classifier Model

D) Training NL Classifier

After designing NN classifier, the system was trained with Quora dataset. We have used Adadelta Optimizer. The use of Adadelta Optimizer eliminates the overhead of tuning learning rate manually. We have also used gradient clipping to avoid vanishing or exploding gradient problem. The training accuracy and cross-entropy loss is shown in figure 2 and 3

The graph in Figure 2 and Figure 3 represents how model loss and accuracy changed over each epoch during training. The model achieved the validation accuracy of 81% at the end of 60th epoch and was stable beyond that. However, the training accuracy continued to rise. This would have resulted in overfitting so, we stopped training the model once validation accuracy was stable. By using mean squared error as the loss function and Adadelta optimizer as the optimizer, the validation loss of 0.13 and training loss of 0.10 was obtained.

IV. RESULT AND DISCUSSION

For evaluating the performance of NN classifier, number of testing on the classifier was performed with the set of test data. The performance evaluation chart for the model is

shown in figure 4. Several model evaluation metrics like Accuracy, Balanced accuracy score, FRR, FAR, Specificity, Precision etc. were used for the evaluation of performance of NL classifier and their adjacent values were calculated as represented in TABLE I. The confusion matrix was prepared after number of tests on the classifier with the testing data. The confusion matrix is shown in TABLE II

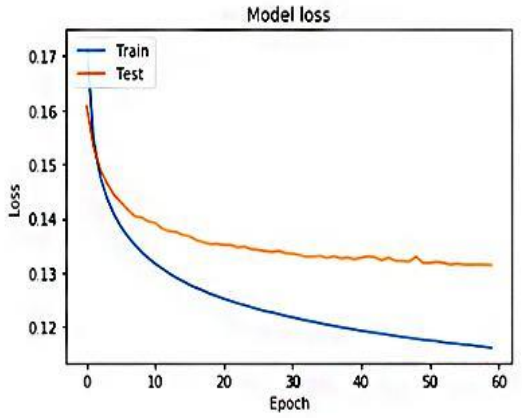

Fig. 2. Loss VS Epoch

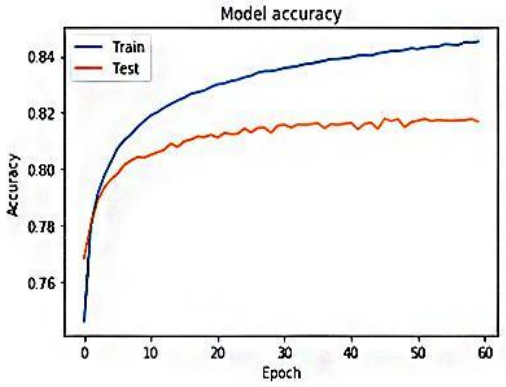

Fig. 3. Accuracy VS Epoch

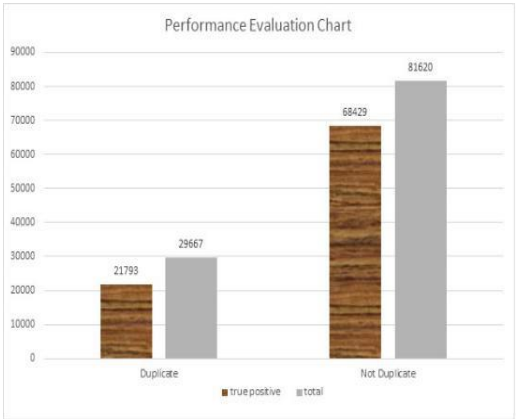

Fig. 4. Performance Evaluation Chart

TABLE I EVALUATION METRICES AND THEIR SCORES

S.N.	Category	Result
1	Accuracy	0.82633
2	FRR	0.3737
3	FAR	0.2932
4	Specificity	0.8968
5	Sensitivity	0.7067
6	F1 score	0.82631
7	Precision	0.8014
8	Recall	0.7067

TABLE II CONFUSION MATRIX

		True Condition	
		Condition Positive	**Condition Negative**
Prediction condition	**Predicted Condition Positive**	True Positive = 31793	False Positive = 7874
	Predicted Condition Negative	False Negative = 13191	True Negative = 68429

V. CONCLUSION

In this paper we represented an approach to better classify the questions. We used Google pretrained Word2Vec model with Keras embedding layer and LSTM to train the classifier. The system gives classification accuracy of 81%.

References

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," The bulletin of mathematical biophysics, vol. 5, no. 4, pp. 115–133, 1943.

[2] H. J. Kelley, "Gradient theory of optimal flight paths," Ars Journal, vol. 30, no. 10, pp. 947–954, 1960.

[3] A. E. Bryson, "A gradient method for optimizing multi-stage allocation processes," in Proc. Harvard Univ. Symposium on digital computers and their applications, vol. 72, 1961.

[4] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[5] T. Mikolov, K. Chen, G. S. Corrado, and J. A. Dean, "Computing numeric representations of words in a high-dimensional space," May 19 2015, uS Patent 9,037,464.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[7] N. Sayer, "Google code archive-long-term storage for google code work hosting," XP055260798, Retrieved from the Internet [retrieved on 20160323], 2014.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] Z. Wang, W. Hamza, and R. Florian, "Bilateral multi-perspective match-ing for natural language sentences," arXiv preprint arXiv:1702.03814, 2017.

[10] C. Saedi, J. Rodrigues, J. Silva, V. Maraev, "Learning profiles in duplicate question detection," in 2017 IEEE International Conference on Information Reuse and Integration (IRI). IEEE, 2017, pp. 544–550.

[11] N. Afzal, Y. Wang, and H. Liu, "Mayonlp at semeval-2016 task 1: Semantic textual similarity based on lexical semantic net and deep learning semantic model," in Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), 2016, pp. 674–679.I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange.