Auto Grader for Short Answer Questions

Pranjal Patil
Department of Civil and Environment Engineering
Stanford University
ppatil@stanford.edu

Ashwin Agrawal
Department of Civil and Environment Engineering
Stanford University
ashwin15@stanford.edu

Abstract

We present a hybrid Siamese adaptation of the Bi-directional Long Short-Term Memory (Bi-LSTM) network for labelled data comprised of pairs of variable length sequences. Our model is applied for the purpose of auto grading of short answer questions. We assess semantic similarity between the provided reference answers and the student response to that particular question. We exceed state of the art results, outperforming handcrafted features and recently proposed neural network systems of greater complexity. For these applications, we provide word embedding vectors to the Bi-LSTMs, which use a fixed size vector to encode the underlying meaning expressed in a sentence (irrespective of the particular wording/syntax). After this the time sequenced output of Bi-LSTM layer is passed through an attention layer to give importance to different words of the sentences. Finally a fully connected layer is proposed to measure the similarity between the word vectors.

Keywords: Sentence Similarity, Attention layer, Bi-LSTM, Fully connected layer

1 Introduction

Short answers are powerful assessment mechanisms. Many real world problems are open-ended and have open-ended answers which requires the student to communicate their response. Consequently, short-answer questions can target learning goals more effectively than multiple choice as they eliminate test-taking shortcuts like eliminating improbable answers. Many online classes could adopt short-answer questions, especially when their in-person counterparts already use them. However, staff grading of textual answers simply doesn't scale to massive classes. Grading answers has always been time consuming and costs a lot of Public dollars in the US. With schools switching to online tests, it is now time that the grading also gets automatic. In order to achieve this we start in this project by tackling the simplest problem where we attempt to make an machine learning based system which would automatically grade one line answers based on the given reference answers.

A typical example of the problem is as below:

Question: You used several methods to separate and identify the substances in mock rocks. How did you separate the salt from the water?

Ref. Answer: The water was evaporated, leaving the salt.
Student -1 Response: By letting it sit in a dish for a day. – (Incorrect)
Student-2 Response: Let the water evaporate and the salt is left behind. – (Correct)

2 Related work

Comparison of sentence similarity is a significant task across diverse disciplines, such as question answering, information retrieval and paraphrase identification. Most early research on measurement of sentence similarity are based on feature engineering, which incorporates both lexical features and semantic features. Research has been carried around WordNet based semantic features detection in the QA match tasks and modelling sentence pairs utilizing the dependency parse trees. However, due to the excessive reliance on the manual designing features, these methods are suffering from high labor cost and non-standardization. Recently, because of the huge success of neural networks in many NLP tasks, especially the recurrent neural networks (RNN), many researches focus on the using of deep neural networks for the task of sentence similarity. [1] proposed a Siamese neural network based on the long short-term memory (LSTM) to model the sentences and measure the similarity between two sentences using Manhattan distance. These models, however, model the sentences mainly using the final state of RNN which are limited to contain all information of the whole sentence. [2] proposed using an attention mechanism to give importance to different words and finally use a fully connected network at the end instead of Manhattan distance.

3 Dataset and Features

We chose the publicly available Student Response Analysis (SRA) dataset. Within the dataset we used the SciEntsBank part of the dataset. This dataset consists of 135 questions from various physical sciences domain. It has a reference short answer and 36 student responses per question. Total size of dataset is 4860 data points. Ground truth labels are available in the dataset whether each student response is correct or incorrect. Data pre-processing including tokenization, stemming and spell checking each of the student responses. We used the Pre-trained Glove embedding trained on Wikipedia and Gigaword 5 with 400K vocabulary and 300 features. We split the dataset as follows: 80% train, 10% validation, 10% test data.

4 Milestone Summary

We divided the auto grading task into 2 parts namely: grading answers of already seen questions given a reference answer and grading answers of unseen questions given a reference answer. The 2^{nd} case is of course more complicated as the algorithm hasn't been trained on the student responses for that question and is only working on the provided reference answer. For the first case [3] showed that k-Nearest Neighbours (kNN) works better than the state of the art neural network approaches.
In kNN approach, we need to decide the weights for the forming the sentence embedding from word embeddings. We came up with the following weights:

$$W_i = IDF_i(W_{pos}(i) - W_{neg}(i))$$

IDF_i = log(Total responses/Responses with i)
$W_{pos}(i)$ = (Correct responses with i / Total correct responses)
$W_{neg}(i)$ = (Incorrect responses with i / Total Incorrect responses)

After getting the weighted sentence vectors, we collected most similar 5 sentences (k=5) from the training set for a particular test sample and assigned the most frequent label. We achieved a 79% accuracy By the above procedure which is quite comparable to state of the are results on this dataset. Although this method is good, it can't be applied to unseen questions as we will not have student responses for that particular question in the train dataset. Hence we decided to take the neural network approach which we feel can generalise text similarity procedure. We observed that correct responses of students are unusually highly correlated and we use this surprising feature in our neural network approach to grade unseen questions.

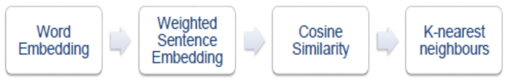

Figure 1: Process for K-NN

5 Methodology

5.1 Framework

Our model composes of two sub-models: sentence modelling and similarity measurement. In sentence modelling we use Siamese architecture consisting of four sub-networks to get sentence representations. Each sub-network also has 3 layers namely: Word-embedding layer, Bi-LSTM layer and an attention layer. In the similarity model, we use a fully connected network and logistic regression layer to compute the correctness of the student response. The complete model architecture is shown in Figure-2.
As mentioned above, from our initial baseline k-Nearest Neighbours (kNN) model, we observed that the correct student responses are unexpectedly highly correlated among each other. We also observed that their correlation among themselves is much higher than with the provided reference answer. Thus we decided to include a couple of correct student responses as well to capture various ways of student writings. The input to our model are 4 sentences, the word sequences of student's response $X_1 = (x_1^1, x_1^2, \dots x_1^n)$ the reference answer provided $X_2 = (x_2^1, x_2^2, \dots x_2^n)$ and the 2 correct student responses $X_3 = (x_3^1, x_3^2, \dots x_3^n), X_4 = (x_4^1, x_4^2, \dots x_4^n)$

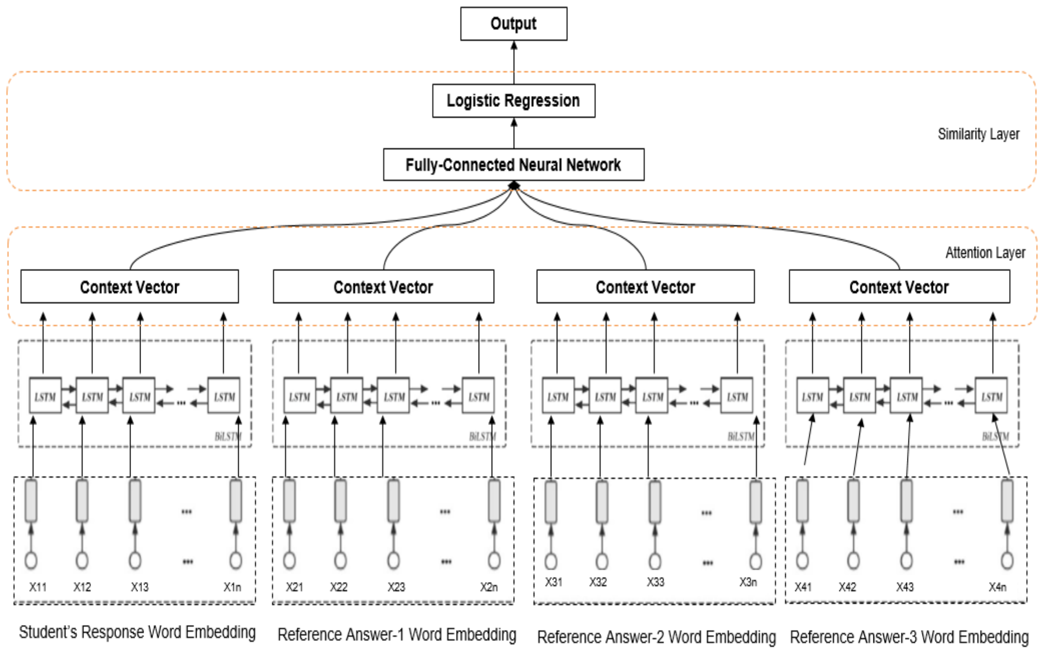

Figure 2: Hybrid Siamese Network

5.2 Sentence Modelling

The sentence modelling part is a process of getting a fixed length sentence vector from individual word vectors. The aim is to get a sentence vector which can help in sentence similarity assessment.

5.2.1 Embedding Layer

The word embedding layer maps every token of the sentence to a fixed length vector. The size of the vector in our model is 300 which are pre-trained GloVe vectors obtained from training over Wikipedia and Gigaword 5 vocabulary.

5.2.2 Bi-LSTM Layer

Take sentence $X = (x_1, x_2 ..., x_T)$ as an example. LSTM updates its hidden state h_t using the recursive mechanism as

$$h_t = sigmoid(WX_t + Uh_{t-1})$$

The LSTM also sequentially updates a hidden-state representation, but these steps also rely on a memory cell containing four components (which are real-valued vectors): a memory state c_t , an output gate that determines how the memory state affects other units, as well as an input (and forget) gate it (and h_t) that controls what gets stored in (and omitted from) memory based on each new input and the current state.

$$i_t = sigmoid(W_i x_t + U_i h_{t-1} + b_i)$$
$$f_t = sigmoid(W_f x_t + U_f h_{t-1} + b_f)$$
$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c)$$
$$c_t = i_t \cdot \tilde{c}_t + f_t \cdot c_{t-1}$$
$$o_t = sigmoid(W_o x_t + U_o h_{t-1} + b_o)$$
$$h_t = o_t \cdot tanh(c_t)$$

where $W_i, W_f, W_c, W_o, U_i, U_f, U_c, U_o$ are weight matrices and b_i, b_f, b_c, b_o are bias vectors.
The Bi-LSTM contains two LSTM: forward LSTM and backward LSTM. The forward LSTM read the sentence from x_1 to x_T , while the backward LSTM read the sentence from x_T to x_1 . We obtain the final vector h_i by concatenating the hidden states of both the layers. Thus a final concatenated vector is passed into the attention layer.

5.2.3 Attention Layer

The attention mechanism can calculate a weight a_i for each word annotation h_i according the importance. The final sentence representation is the weighted sum of all the word annotations using the attention weight.

$$e_i = tanh(W_h h_i + b_h)\ e_i \in [-1, 1]$$
$$a_i = \frac{exp(e_i^T u_h)}{\Sigma exp(e_t^T u_h)}$$
$$r = \Sigma a_i h_i$$

5.3 Similarity Measurement

The similarity measurement model functions as a binary classifier for the learned sentence embedding. Our model is an end to end model which means that sentence modelling layer and the similarity measurement model can be trained together.

Fully Connected Layer: Each output of our sentence modelling layer is a fixed size vector. We pass each of the student response , reference answer pair into the fully connected layer to measure the similarity between them. In this way we have 3 fully connected layers outputting 3 vectors for the pair wise similarity with the student response. [2] showed that this works much better that Manhattan distance which was used by [1]

Logistic Regression Layer: The output of the fully connected layer is taken as input by this layer and it outputs a probability for the student response being correct.

5.4 Assesment and Loss Function

To evaluate the performance of our model, we chose two metrics namely accuracy (ACC) and mean square error (MSE). A threshold of 0.5 is used on predicted probability for assigning the final labels. For each sentence pair, the loss function is defined by the cross-entropy of the predicted and true label for training:

$$Loss = \text{ylog}(\tilde{y}) + (1-y)log(1-\tilde{y})$$

where y is the true label and $\tilde{y}$ is the output probability for correct response.

It is most easily interpret able as well as an apt choice for our task which is very similar to a classification task.

6 Experiments and Results

The hyper parameters used in our model were adapted from [1] as it acted baseline for our case. *Number of hidden layers in LSTM = 50*. We selected the number of time-steps for the LSTM model to be equal to the length of the largest sentence in training the set which required us to pad the rest of the sentences to make the length equal. We used Adam optimizer to achieve faster convergence. Convergence rate was not an issue right now, but we wanted to make the model future proof for when we would run this on a much larger dataset than the current one.

Each model was trained for 50 epochs and batch size 16. Softmax activation function was used in the attention layer. LSTM was initialized with normal weights. Though [1] suggests that LSTM is highly sensitive to initialization and we should start from a pre-trained network, we initialized the parameters randomly due to time constraints. L2 regularization was used in LSTM layer.

We built various models permuting with CNN, LSTM , Bi-LSTM, Attention layer, FNN and Manhattan Distance. Some of our best results are summarised in the table below. Our model is the best result obtained from these and it's architecture has been described above.

Model	Accuracy(%)	MSE
LSTM + Manhattan Distance	62%	0.25
LSTM + Attention + FNN	73%	0.18
CNN + Bi -LSTM + Manhattan	69%	0.20
OurModel	76%	0.16

Table 1: Results

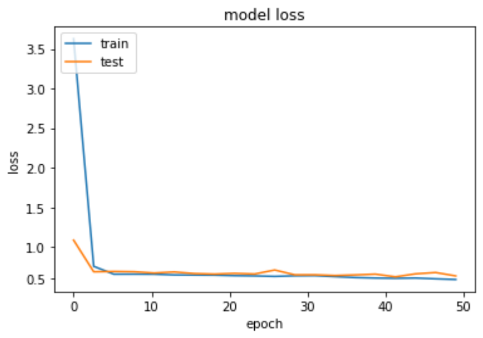

Figure 3: Loss Curve

7 Discussion

Our Hybrid Siamese model achieved the highest accuracy. The credit for this success can be given to the kNN intuition. The observation that correct answers given by student are very similar to correct answers given by other students has helped in achieving this increased accuracy. Also it can be seen that the attention layer creates a large increase in the accuracy of the models as compared to the ones without accuracy. The ability of the attention layer to identify the weightage of each word according to its importance in the reference answer. We studied the cases in which the model was misclassifying the student answers. We found that there were two main causes for misclassification.

7.1 Length of student answer

The model misclassified cases where the difference between the length of the student answer and reference answers was large. We tried to overcome this by replacing the final fully connected layer with a cosine similarity measuring layer. This led to lower accuracies. Therefore the fully connected layer is better than cosine similarity but we need to change some properties of the layers to get better results. We believe that this problem can be solved by using a different attention layer which will enable the algorithm to remember the important words for longer time intervals.

7.2 Issue of Key words

Next up we observed the model misclassified answers which were missing the keywords from the reference answers. These student answers seem similar to the reference answer when we read it but are misclassified by the algorithm. This could be a result of the attention layer giving extra weight to the keywords and not being able to identify a phrase which means the same as the keyword. The example of the same is shown below.

Modifications must be implemented in the attention layer such as changing the activation etc to make it more robust.

Question : What is the relation between tree rings and time?

Ref Answer: As time increases, number of tree rings also increases.
Student Answer: They are both increasing

Original Label: Correct

Model Result: Incorrect

8 Conclusion & Future Work

Our hybrid model with the intuition of kNN beat all the other models on our dataset. Building upon our learnings from this project we would like to expand the analysis by training on run it on a larger unseen and out of domain dataset to gauge its robustness. During the poster presentation we talked with a researcher who was interested in providing us with a much larger dataset. We would also like to address all the issues we observed with our current model. We will be trying out different attention layer to smooth out key word issue. We would also consider adding better reference answers or better similarity detection mechanisms in the future.

9 Contributions

We worked on each part collaboratively and didn't explicitly divided the tasks. We both had equal contributions to literature review, data collection, writing code and report preparation.

10 Github Link

This is the link to our code in Github repository:

Click here to access the Github Code

References

[1] Jonas Muller and Aditya Thygarajan, “Siamese Recurrent Architecture for learning sentence similarity”, AAAI-16

[2] Ziming Chi and Bingyan Zhang, “A sentence similarity estimation method based on improved Siamese Network”, JILSA-2018

[3] Tianqi Wang et.al, “Identifying Current Issues in Short Answer Grading”, ANLP-2018

[4] Brain Riordan et.al, "Investigating neural architectures for short answer scoring", ACL-2017

[5] Grenfenstette, E., et,al, "Reasoning about entailment with Neural Attention",arXiv: 1509.06664

[6] Chen,Q. et.al, "CAN: Enhancing Sentence Similarity Modeling with Collaborative and Adversarial Network.", ACM SIGIR-2018