Real-time Automated Answer Scoring

Akash Nagaraj
Department of Computer Science
PES University
Bangalore, India
akashn1897@gmail.com

Mukund Sood
Department of Computer Science
PES University
Bangalore, India
mukundsood2013@gmail.com

Dr. Gowri Srinivasa
Department of Computer Science
PES University
Bangalore, India
gsrinivasa@pes.edu

***Abstract*— In recent years, the role of big data analytics has exponentially grown, and is now slowly making its way into the education industry. Several attempts are being made in this sphere in order to improve the quality of education being provided to students while simultaneously improving teaching techniques to reduce the burden put on the teacher. While the implications of such a collaboration have in fact been carried out before, automated scoring of answers has been explored to a rather limited extent.**

The biggest obstacle to choosing constructed-response assessments over the traditional multiple-choice assessments is the large cost, and effort that comes with their scoring, and this is precisely the issue that this project aims to solve.The aim is to accept raw-input from the student in the form of their answer, pre-process these answers if required, and automatically score the answers. Evaluation of these answers will be based purely on the previous examples on which the model has been trained. In addition, we have made this a real-time system that captures "snapshots" of the writer's progress with respect to the answer, allowing us to see how the student has arrived at his/her final answer. This additional information can allow us to analyse where students err, allowing us to better understand the thought-process of students as they take tests and possibly unearth previously unknown trends.

I. INTRODUCTION

Automated answer scoring is the use of specialized computer programs to assign grades to answers written in an educational setting. It is a method of educational assessment and an application of natural language processing.

Attempts to build an automated grading system dated back to 1966 when Ellis B. Page proved on The Phi Delta Kappan that a computer could do as well as a single human judge [1]. Since then, much effort has been put into building the perfect system. Intelligent Essay Assessor (IEA), developed by Peter Foltz and Thomas Landauer, was first used to score essays for large undergraduate courses in 1994 [2]. The automated reader developed by the Educational Testing Service, e-Rater, used hundreds of manually defined features. It was trained on 64 different prompts and more than 25,000 answers on a 6-point scale from 1 to 6. Evaluated on the quadratic weighted kappa calculated between the automated scores for the answers and the resolved score for human raters on each set of essays, e-rater could only achieve a kappa score below 0.5. [3]

In 2012, the Hewlett Foundation sponsored a competition on Kaggle called the Automated Student Assessment Prize (ASAP). The competition also used quadratic weighted kappa to measure the similarity between the human scores and the automated scores. 154 participants attempted to predict the essay score. The winning team got a kappa score of 0.81407. [4] Later, a team at Carnegie Mellon University built a model using dense and sparse features, trained on the same dataset to achieve the kappa score of 0.833. [5]

Our approach will be different from the previous attempts to solve the problem, using deep learning, as opposed to domain oriented predefined features. The best way to improve one's own writing skill is to write an answer, receive feedback from your instructor, and based on the feedback, revise your answer. Repetition of this process as often as possible is advised up until an optimal score is reached. However, the problem arises in the fact that continuously evaluating these essays requires a lot of time and puts an enormous load on the classroom teacher who is often found in a position where she must provide feedback to a possible 30 essays or more each time a topic is assigned. As a result, teachers are not able to give writing assignments as often as they would wish. Keeping this in mind, researchers have sought to develop applications that automate essay scoring and evaluation, with the goal of allowing teachers more flexibility in designing a course so as to maximize the learning of the student without having to compromise due to a large work load. Automated scoring of answers is not meant to serve as a replacement to a human evaluator, but, merely as an aid to the evaluator.

II. RELATED WORK

A. Recurrent Neural Networks for Language Understanding [6]

Shows the effectiveness of Recurrent Neural Networks for language understanding. The core of their project was to take words as inputs in a standard RNN-LM and to predict slot labels instead of the traditional method of words on the output side. The RNN-LM represents each word as a high-dimensional real-valued vector in such a way that similar words tend to be closer together and relationships between words are preserved. They demonstrated the use of the Elman Architecture and using the ATIS Dataset, they were able to show that the RNN outperformed the Conditional Random Fields (CRF) approaches by a large margin as well as previous neural-net approaches.

B. Automated Essay Scoring Using Bayes' Theorem [7]

Bayesian Models for text classification from the information science field were extended and applied to student produced essays, with each model calibrated to 462 essays with no score points. They manipulated some variables, such as trimming, stemming and the use of stopwords to imporve their accuracy. They were able to achieve an accuracy of only 80%. The major drawbacks of this paper are that it uses a very simple Bayesian Model for the classification.

C. Cosine similarity to determine similarity measure: Study case in online essay assessment [8]

Implementation of the weighting of Term Frequency - Inverse Document Frequency (TF-IDF) method and Cosine Similarity with the measuring degree concept of similarity terms in a document. Tests have been carried out on a number of Indonesian text-based documents that have gone through the stage of pre-processing for data extraction purposes. This process results is in a ranking of the document weight that have closeness match level with expert's document.

III. PROBLEM STATEMENT

The aim is to implement a system that evaluates a student's answer A and recommends a score $s(A)$, accurately as well as efficiently, such that; rather than just evaluating the end-product A_f, consider a sequence of intermediate answers A_1, A_2, ..., A_n (where A_n = A_f) captured at suitable times t_1, t_2, ..., t_n as the student builds his/her answer.
By relating $s(A_i)$ vs t_i and plotting the graph, we will be able to gain insight on the thought process and steps taken by which the student arrived at his/her answer. For instance, two students may end up with the same score but one of the students may have gradually built up to this score, whereas the other may have actually written an answer that earned a higher score before making modifications that lowered the score.

A. The Dataset

The data set we are using for this project is The Hewlett Foundation: Automated Essay Scoring [4].

About the data set:

- There are 8 different sets of essays, each generated from a single prompt.
- Selected essays range from an average length of 150 to 550 words per response.
- Each essay was hand-graded a human evaluator, which means we are using real-world data to produce real-world results.
- There are 10686 samples in total. We used a 85:15 data split for training and testing, as well as validation.
- Each set has a different grading scale.

B. Evaluation Metric [16]

In recent years, the role of big data analytics has exponentially grown, and is now slowly making its way into the education industry. There is a growing need to tap into the massive reserves of education data emanating from both private as well as government institutions. Several attempts are being made in this sphere in order to improve the quality of education being provided to students while simultaneously improving teaching techniques to reduce the burden put on the teacher. While the implications of such collaboration have in fact been carried out before, automated scoring of answers has been explored to a rather limited extent.

The biggest obstacle to choosing constructed-response assessments over the traditional multiple-choice assessments is the large cost, and effort that comes with their scoring, and this is precisely the issue that this project aims to solve.The project is to accept raw-input from the student in the form of their answer, pre-process these answers if required, and automatically score the answers. Evaluation of these answers will be based purely on the previous examples on which the model has been trained. In addition, we have made this a real-time system that captures "snapshots" of the writer's progress with respect to the answer, allowing us to see how the student has arrived at his/her final answer. This additional information can allow us to analyse where students err, allowing us to better understand the thought-process of students as they take tests and possibly unearth previously unknown trends.

$$\kappa = 1 - \frac{\sum_{ij} W_{ij} O_{ij}}{\sum_{ij} W_{ij} E_{ij}} \tag{1}$$

- The Weight matrix W is calculated based on the difference between raters scores.
- O_{ij} corresponds to the number of answers that received a rating i by Rater A and Rating j by Rater B.
- E is the nxn histogram matrix of expected ratings.

The evaluation metric used for the project is quadratic weighted kappa [10]. The Quadratic Weighted Kappa (QWK) metric typically varies from 0 (only random agreement between raters) to 1 (complete agreement between raters), and doesn't penalize for an incorrect score match, but rather, it takes into account the error in the scores, with respect to the range of scores. Therefore, it serves as a much better evaluation metric as compared to accuracy, since in our problem, we're primarily comparing inter-rater agreement.

IV. APPROACH

The primary objective is to build a model that accepts an answer as input and automatically outputs the score of that answer, in an efficient manner that can be done in *real-time* as the student writes the answer. The score would be evaluated at the end of sentences, words or ever a few systems that only perform the analysis and evaluate the final draft in a answer.
Coming to the pre-processing don. What we first did, was try to build a Bag of Words Model and examine the results it would give us, but they were not very promising.

TABLE I
RANDOM FOREST V/S SVM RBF

	Word2Vec	
Model	**QWK**	**Dim**
Random Forests	0.9535	300
Random Forests	0.9563	200
Random Forests	0.9508	100
SVM Radial Basis Function	0.9619	300
SVM Radial Basis Function	0.9568	200
SVM Radial Basis Function	0.9509	100

TABLE II
LSTM V/S DNN V/S COMBINED MODEL

	Word2Vec	GloVe	
Model	**QWK**	**QWK**	**Dim**
Long Short Term Memory	0.9653	0.9566	300
Long Short Term Memory	0.9642	0.9623	200
Long Short Term Memory	0.9512	0.9433	100
Deep Neural Network	0.9643	0.9555	300
Deep Neural Network	0.9576	0.955	200
Deep Neural Network	0.9611	0.9442	100
Combined Model	0.9721	-	100

Based on our findings, we decided to move forward and remove all stop-words from each of the essays being passed to our Model. We also implemented a rudimentary Spell Check on the essays based on the Peter Norvig Spell Check Algorithm and stemming, however, the results we achieved were substandard possibly owing to the fact that human evaluators would consider a misspelled word as an important factor (reducing overall marks of the answer) and we decided against using them continuing forward with our research. The interesting bit of the pre-processing comes, where we decided to implement a word-embedding model. As mentioned above we had two possible choices, GloVe [12] and Word2Vec [16]. Based on some research into the capabilities of both models we found that the Word2Vec would possibly be a better model, however, just to confirm for ourselves, we decide to implement both, working however, on the assumption that the Word2Vec Model would outperform the GolVe Model.

The role that Word2Vec plays in our implementation is a massive one. Developed by team of researchers at Google, it is an extremely useful group of related models to produce word vectors of your input. The Word2Vec Model is trained on the input specified, and is trained in such a way that words that have a similar meaning have a smaller vector distance between them. This representation of words is very helpful going forward with our predictions primarily because we are able to train our Model based on the type of input, rather than predefined meanings of the word, which was consequently one of the reasons that we chose this model over GloVe, thereby ensuring that words that have a similar meaning in that context or tend to appear together frequently have smaller distances than words that are similar by their intrinsic meaning.

Once we obtained the vector representations of our input, we used Vector Averaging to get a specific Vector Value for each essay in the dataset, which we could then use to make better predictions. In our approach we used four Models; a Random Forest Classifier, a Radial Basis Support Vector Machine, a Deep Neural Network and a Short Long Term Memory Unit which is a form of a Recurrent Neural Network. The Random Forest Classifier [17] and the Support Vector Machine [18] returned very good results. Looking into Random Forest Classifiers, they are an example of an ensemble learner built on decision trees, is a direct consequence of the fact that by maximum voting from a panel of independent judges, we get the final prediction better than the best judge.

Our Deep Neural Network is a simple two-layered feed forward neural network, where we use the 'Rectified Linear Units' as the activation function for the first layer and a 'Softmax' activation function [19] for the output layer to squash the output vector. We tried multiple activation functions, but verified that these activation functions give us an optimum result and the loss function that we have used is 'sparse categorical crossentropy' since our problem deals with more than two classes. While building the Word2Vec Model we achieved the best results when the dimension for each word vector was set to 300. Once trained, the Neural Network architecture would be used to evaluate 'snapshots' of the students' intermediate responses which will give us a graph of the student's score vs time intervals, which could answer many interesting questions with respect to the student's performance.

Recurrent networks take as their input not just the current input example they see, but also what they have perceived previously in time. The decision a recurrent net reached at time step t-1 affects the decision it will reach one moment later at time step t. Hence, RNNs have two sources of input, the present and the recent past, which combine to determine how they respond to new data. This quality of an RNN is what allows it to produce robust models when it comes to textual input, because of its sequential quality. The sequential information is preserved in the RNN's hidden state, which manages to span many time steps as it cascades forward to affect the processing of each new example. It is finding correlations between events separated by many moments, and these correlations are called long-term dependencies, because an event downstream in time depends upon, and is a function of, one or more events that came before.

In the mid-90s, a variation of recurrent net, called Long Short Term Memory units (LSTM) was proposed as a solution to the vanishing gradient problem [24]. LSTMs help preserve the error that can be back-propagated through time and layers. By maintaining a more constant error, they allow RNNs to continue to learn over many time steps (over 1000), thereby opening a channel to link causes and effects remotely. This feature of an LSTM, aids the RNN in the

model we have built to allow for better predictions of each answer.

Now, considering that we are trying to predict a grade for a student, a simple accuracy test where we compare the number of correct predictions of our model, would not be a very good metric, therefore, as specified in the source of the dataset, have used Quadratic Weighted Kappa as our Evaluation Metric. Quadratic Weighted Kappa allows us to calculate the agreement between evaluator, which in our case would be the agreement of the score predicted by our models, against the score predicted by an evaluator whose scores we are using to train the model. This turns out to be a model far more similar to a real-world scenario where agreement between raters is considered, taking into account the weighted errors.

Finally, we used a method similar to Weighted Average. We obtained the predictions from each Model that we built, namely the LSTM Mode, the Deep Neural Network Mode, the SVM Radial Basis Function Model and the Random Forrest Classifier Model, along with their respective Quadratic Weighed Kappa values, and applying the formula:

$$\kappa_f = \frac{\kappa_1 * S_1 + \kappa_2 * S_2 + \kappa_3 * S_3 + \kappa_4 * S_4}{\sum_{i=1}^{4} \kappa_i} \tag{2}$$

we were able to obtain better predictions for each essay that resulted in an overall higher Quadratic Weighted Kappa Score.

A. General Classifier Results

Table 2 above shows us the QWK of the Random Forrest Classifier and the SVM using a Radial Basis Function. The high QWK was unexpected, however we attribute it towards the pre-processing done as well as the use of Word Vectors.

V. EXPERIMENTS AND RESULTS

Based on the above approaches, we ran multiple tests to figure out which Model would give the best results. We documented all the results and we now provide justifications for each of them, along with a couple other pre-processing techniques that we used, and how they fared as well. Earlier, we had justified our selection of Word2Vec over the pre-trained GloVe Vectors, but as a confirmation, we ran tests on both models just to take a look at how they performed under the same conditions.

A. Effect of Pre-processing

We used a couple of different pre-processing techniques which affected our results, so we feel it is appropriate to discuss these before our final Results

1) Stemming [20]: Stemming is the process of reducing inflected words to their word stem, base or root form. When we applied this method we incurred a drop in the QWK across all models. The reason behind this drop lies in the importance of the way in which an answer is written by a student. A human scorer would evaluate students who have better grammar and use of words higher than those who did not. Hence, when we tested the model after stemming the training and testing sets, we incurred a considerable drop in accuracy.

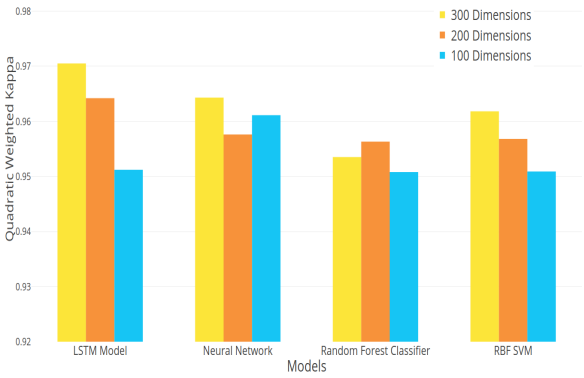

Fig. 1. Comparison of the different models

2) Spell-Correction [11]: To implement Spell-Correction we used Peter Norvig's Spell Check Algorithm [11], and this method as well, similarly to Stemming, resulted in a drop of accuracy. The reason for this is identical to the one listed above, and hence, we deiced not to use either of these methods moving forward with our testing.

B. Neural Network Results

Table 1 above, has a comparison of how the LSTM Model and Deep Neural Network Model worked with respect to both Word Embedding Models. Based on the results, the following two observations can be made:

1) Word Embedding Model: You can see that in general the Word2Vec Model outperforms the GloVe Model. This is because Word2Vec was trained on our own dataset whereas the GloVe Model used pre-trained Word Vectors. This phenomenon is best explained with an example. Let's say we have two words, *threads* and *string*, in terms of Computers (which was the topic of most of the answers in our dataset) they have a completely different meaning to what they do in day-to-day life. Hence, while predicting a score, the Word2Vec Model allows us to use the relevant word in the relevant situation which GloVe does not, leading to a slightly lower Quadratic Weighted Kappa.

2) Prediction Model: We see that the LSTM outperforms the DNN, this increase in QWK can be attributed towards the fact that an LSTM is basically a Recurrent Neural Network, this means that it works best with sequential data, and in our case, text is a perfect example of sequential data. Additionally, an LSTM has a memory unit which allows it to take into account a larger context, thereby giving slightly better results. However we see that the Combined Model yields the best overall results, due to the fact that it takes into account the prediction made by each Model and gives a weighted average that tends to be the best fit for the Answer in question.

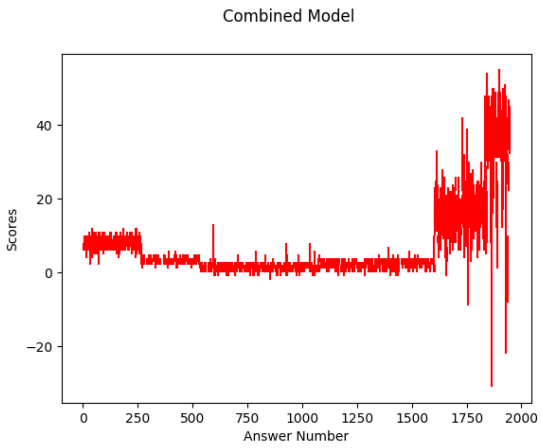

Fig. 2. Errors of the Combined Model using Word2Vec

C. Graphs

The visualizations seen is a general error graph of the predictions obtained using the Combined Model.
Finally you can see our results compared to the previous best results of the competitors who participated in Kaggle's competition.
We also added a graph of QWK v/s the Model used which is a comparison of all our Models compared to previous results. As illustrated in the graph, our Model has exceeded previous prediction Model by a large margin.

D. Optimizers

With regard to optimizers, we used a numerous predefined optimizers such as $StochasticGradientDescent$ [21], $Adam$ [22], $Adagrad$, $AdaDelta$ [23] etc. But $RMSProp$ yielded the best results, with a Quadratic Weighted Kappa of about 0.974

E. Real-Time System

We implemented our Real-Time system using a web-server designed on python's flask web framework. We grade the scores of students as they type at intervals of 0.66 seconds based on studies that show this to be the typing speed per word of the average human. The Real-time system predicts the score of "snapshots" of a student's answer using the combined model in about 0.3 seconds (running on a Intel i7-7820 Processor), making it an interactive system.

VI. Conclusion

The inspiration behind this project was to make the job of a teacher easier. The crux of a formal education is written assignments, and teachers often are not able to give as many as they would like. Using a system built around Automated Scoring, we could move into an era where teachers can Model courses however they wish without being restricted by their workload.
We also find that this project could be helpful in getting to know how a student answers particular questions, and in what way they first approach a particular problem. This would give us a huge amount of data with respect to how students think, this giving us a better idea on how to teach certain concepts to students, in ways that they best understand.

References

[1] Page, Ellis B. The imminence of... grading essays by computer. The Phi Delta Kappan 47.5 (1966): 238-243..
[2] Foltz, Peter W., et al. Implementation and applications of the Intelligent Essay Assessor. Handbookof automated essay evaluation (2013): 68-88.
[3] Attali, Yigal, and Jill Burstein. Automated essay scoring with e-rater V. 2. The Journal of Technology, Learning and Assessment 4.3 (2006).
[4] Kaggle. Develop an automated scoring algorithm for student-written essays. (2012). https://www.kaggle.com/c/asap-aes
[5] Robertson, Stephen. Understanding inverse document frequency: on theoretical arguments for IDF. Journal of documentation 60.5 (2004): 503-520.
[6] Yao, Kaisheng, Geoffrey Zweig, Mei-Yuh Hwang, Yangyang Shi, and Dong Yu. "Recurrent neural networks for language understanding." In Interspeech, pp. 2524-2528. 2013.
[7] Rudner, Lawrence M., and Tahung Liang. "Automated essay scoring using Bayes' theorem." The Journal of Technology, Learning and Assessment 1, no. 2 (2002).
[8] Lahitani, Alfirna Rizqi, Adhistya Erna Permanasari, and Noor Akhmad Setiawan. "Cosine similarity to determine similarity measure: Study case in online essay assessment." In Cyber and IT Service Management, International Conference on, pp. 1-6. IEEE, 2016.
[9] Foltz, Peter W., Darrell Laham, and Thomas K. Landauer. "Automated essay scoring: Applications to educational technology." In World Conference on Educational Multimedia, Hypermedia and Telecommunications, vol. 1, pp. 939-944. 1999.
[10] Cohen, Jacob. "Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit." Psychological bulletin 70, no. 4 (1968): 213.
[11] Norvig, Peter. "Natural language corpus data." Beautiful Data (2009): 219-242.
[12] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543. 2014.
[13] Ramos, Juan. "Using tf-idf to determine word relevance in document queries." In Proceedings of the first instructional conference on machine learning, vol. 242, pp. 133-142. 2003.
[14] Sepp Hochreiter and Jrgen Schmidhuber, Long Short-Term Memory, Neural Computation Volume 9, Issue 8, November 15, 1997 p.1735-1780
[15] Mikolov, Tomas; et al. "Efficient Estimation of Word Representations in Vector Space". https://arxiv.org/abs/1301.3781
[16] Mary L. McHugh, Inter-Rater Reliability: The Kappa Statistic.
[17] Breiman, L. Machine Learning (2001) 45: 5. https://doi.org/10.1023/A:1010933404324
[18] M.A. Hearst ; S.T. Dumais ; E. Osuna ; J. Platt ; B. Scholkopf, Support Vector Machines Published in: IEEE Intelligent Systems and their Applications (Volume: 13, Issue: 4, July-Aug. 1998)
[19] R. A. Dunne and N. A. Campbelly, "On The Pairing Of The Softmax Activation And Cross Entropy Penalty Functions And The Derivation Of The Softmax Activation Function" 8th Aust.Conf.on Neural Networks, Melb.1997
[20] Julie Beth Lovins, Development of a Stemming Algorithm, [Mechanical Translation and Computational Linguistics, vol.11, nos.1 and 2, March and June 1968]
[21] Antoine Bordes, Lon Bottou, Patrick Gallinari, SGD-QN: Careful Quasi-Newton Stochastic Gradient Descent Journal of Machine Learning Research 10 (2009) 1737-1754 Submitted 1/09; Revised 4/09; Published 7/09
[22] Diederik P. Kingma, Jimmy Ba, Adam: A Method for Stochastic Optimization
[23] Matthew D. Zeiler, ADADELTA: An Adaptive Learning Rate Method
[24] Y. Bengio, P. Simard, P. Frasconi: Learning long-term dependencies with gradient descent is difficult