Automated Essay Scoring as Basic Regression

Ashesh Singh
CS421 Project Report
asing80@uic.edu

***Abstract*—Automated evaluation of essays apply mechanisms that attempt to match the grading aptitudes of humans. Such systems are an active research agenda for several institutes as well as standardized testing agencies. In this paper, I show the outcome of using a simple regression-based approach for accomplishing this task. The results are achieved by training a sequential model in Tensorflow to produce continuous values. I test several writing features taken individually and in combination to show their contribution towards the goodness of each trained model.**

***Index Terms*—grading, regression, feature extraction, kappa**

I. Introduction

Performance on essays is interpreted as vital for measuring students' understanding of a subject. For a long time essays were an integral part of examinations at all educational levels right from elementary school to university [1]. However, in recent times such questions that require long-form text response by the student have fallen out of favor [1] [2]. One major reason for this is the sheer amount of human capital required to grade them. For example, grading a single high school English essay can take anywhere from fifteen to thirty minutes [3]. When we compare this to other types of questions: multiple-choice, true or false, fill in the blanks; all of which can be graded within seconds or even automated through Optical character recognition (OCR) technology, it's easy to see the negative bias. This is the motivation behind Automated Essay Scoring (AES) systems that can grade the essays without any human involvement and hence save on the capital required.

Such systems play a pivotal role in large scale standardized assessments like the Test of English as a Foreign Language (TOEFL) and Graduate Record Examinations (GRE) which require grading of millions of essays in a short time [4]. The testing agency behind these, Educational Testing Service (ETS) employs its commercial, closed source AES implementation named *e-rater*. The *e-rater* engine is touted to provide a holistic score for an essay based on grammar, usage, mechanics, style, and organization, and development. This feedback is based on natural language processing research specifically tailored to the analysis of student responses and is detailed in ETS's research publications [5]. However, the human component is not totally absent during the process since the essays are first graded by *e-rater* and a human independently. In case of a huge difference between the ratings, a second human rater is approached to make a decision. This suggested that although such systems can be used in a commercial setting, their reliability can not be taken for granted. Nevertheless, AES systems like *e-rater* and others have evolved significantly over the years to include more nuanced essay features that have helped them mimic a human-grade more closely [5] [6].

II. Related Work

AES can be approached as a regression or classification problem. When modeling the task as a classification problem, the scores are treated as labels and algorithms like K-nearest neighbors (KNN), Logistic Regression, Support Vector Machines (SVM) can be applied to predict these labels. On the other hand, in a regression approach, we focus on predicting a continuous set of values as output. This works even if the input training scores are on discrete levels since the predicted values can be rounded off. Further, AES implementation can be done through purely statistical, machine learning or deep learning models. Having a high accuracy is of cardinal importance for an AES. As a result, a large number of publications focus on deep learning models since they provide high accuracy at the expense of time spent on model training, and computational resources required. More recent work has focused on Memory-Augmented Neural Model that deal with complex reasoning Natural Language Processing (NLP) tasks and have been shown to outperform Long short-term memory (LSTM) on some complex reasoning tasks [7].

Regardless of the specific implementation approach, the design for most AES systems follows a fixed pattern involving two processes: learning and prediction. First, the essay texts are processed through an NLP module that extracts several predefined features from each essay. These features could range from very basic (eg. essay length, average sentence length, average word length) to more sophisticated ones (eg. number of grammatical and spelling errors, sentence coherence, parts of speech (POS) tags, count of noun phrases). The NLP module may additionally perform certain operations for optimal performance. These features are learned to create a model by the learner module and are further used by the predictor module to score un-scored new essays [7].

In this report and the accompanying project, I use a simple linear regression approach to make an AES system. Other advance multiple regression-based techniques have also been used in the past for this purpose and their results are well-documented [7] [8].However, I chose this alternative approach because of its simplicity and since it allows me to quickly analyze the effect of several essay features on the final trained model. Linear Regression also performs reasonably well considering the level of difficulty in implementation and the computing resources required. I am interested in observing what features taken individually and in sets give a good model for AES. This is accomplished by first extracting those features and then training several models for different feature combinations and comparing their results against a common evaluation metric Quadratic Weighted Kappa (QWK) which is discussed in later sections.

III. Methodology

This sections provides an overview of the dataset, feature extraction and the model training steps.

A. Dataset

Building an AES system requires a significant amount of scored essay data. The dataset used in this project is from the 2012 Automated Student Assessment Prize (ASAP) Kaggle[1] competition sponsored by the William and Flora Hewlett Foundation (Hewlett) [2]. The dataset was made available in several formats and on average, each essay is approximately 150 to 550 words in length, written by students ranging in grade levels from Grade 7 to Grade 10. Each of the ASCII formatted essays belongs to one of 8 sets and has one or more corresponding human scores in several domains. The dataset files contain 28 columns, some of them are described below:

- `essay_set`: 1-8, an id for each set of essays
- `essay`: The ascii text of a student's response
- `rater1_domain1`: Rater 1's domain 1 score; all essays have this
- `rater2_domain1`: Rater 2's domain 1 score; all essays have this
- `domain1_score`: Resolved score between the raters; all essays have this

It is important to note that each of these essay sets has a different prompt and is scored in varying ranges as depicted in *Table I*. This difference becomes important for developing prompt specific AES systems that are tuned to work differently for varying types of writing prompts as compare to generic AES implementations that use the same internal functions to score all essays. The dataset files have extensive details on the specific prompts and scoring rubrics which are briefly discussed here:

- Essay Set 1: Students are required to provide a well-developed response that takes a clear and thoughtful position on advances in technology (computers) and provides *persuasive* support.
- Essay Set 2: Students are required to exhibit a superior command of language skills while providing *persuasive* support for their position in response to the topic of censorship in libraries.
- Essay Set 3: Students are required to give *source dependent responses* that demonstrated an understanding of the text by explaining certain details.
- Essay Set 4: Students are required to give *source dependent responses* that demonstrated an understanding of the text by explaining the author's conclusion of the text.
- Essay Set 5: The students' *source dependent responses* is expected to be a clear, complete, and accurate description of the mood created by the author of the text.
- Essay Set 6: The students' *source dependent responses* is expected to be a clear, complete, and accurate description of the text by explaining certain details.
- Essay Set 7, 8: The students' *narrative/expository responses* is expected to be clearly focused on the topic and thoroughly developed with specific, relevant details

B. Feature Extraction

Feature extraction is critical for making any learning model. Knowing what features to extract from some data requires a deep understanding of that data and preferably some domain knowledge to assist in the selection process [11]. Lacking this, I have relied on extracting only some basic text properties which I feel should give reasonable results. These Features are roughly categorized into three sets, each of which is listed below and in TableII.

1) Meta Features:
   a) Essay Length: The count for number of words in an essay text.
   b) Average Sentence Length: The average number of sentences in an essay text.
   c) Average Word Length: The average number of characters in words of an essay text.
2) Grammar Features
   a) Sentiment: Identifies positive, negative and neutral emotions in essay text.
   b) Noun Phrases: The count of text phrases that has a noun as its head.
   c) Syntax Errors: The count of spelling, and grammatical errors in the essay.
3) Redability Features
   a) Automatic Redability Index: Tries to gauge the understandability of a text based on factor of characters per word [12]. Calculated using the formula:

$$0.5\left(\frac{words}{sentences}\right)+4.71\left(\frac{characters}{words}\right)-21.43$$

   b) Difficult Words: Number of difficult unique words used within the essay.

These features are fairly easy to obtain through existing software libraries like `textblob`[2], `language-check`[3], and `textstat`[4].

C. Model Training

I used the `tensorflow`[5] platform to build a Sequential[6] model with two densely connected hidden layers, and an output layer that returns a single, continuous value. Further,

[1] https://www.kaggle.com/c/asap-aes/data

[2] https://textblob.readthedocs.io/en/dev/api_reference.html

[3] https://pypi.org/project/language-check/

[4] https://pypi.org/project/textstat/

[5] https://www.tensorflow.org/

[6] https://www.tensorflow.org/api_docs/python/tf/keras/Sequential

TABLE I
STATISTICS ON RESOLVED SCORE OF ASAP ESSAY SETS

essay_set	1	2	3	4	5	6	7	8
count	1783	180000	172600	177000	180500	180000	156900	72300
mean	8.52	3.41	1.85	1.43	2.41	2.72	16.06	36.95
std	1.53	0.77	0.81	0.94	0.97	0.97	4.58	5.75
min	2	1	0	0	0	0	2	10
25%	8	3	1	1	2	2	13	33
50%	8	3	2	1	2	3	16	37
75%	10	4	2	2	3	3	19	40
max	12	6	3	3	4	4	24	60

TABLE II
EXTRACTED FEATURES FOR RANDOM ESSAYS

essay_length	avg_sentence_length	avg_word_length	sentiment	noun_phrases	syntax_errors	readability_index	difficult_words
231	16.35	4.47	0.08	12	12	11.0	26
23	23.00	4.60	0.00	1	0	12.0	5
43	14.33	4.39	0.03	2	2	6.8	5
411	21.47	4.99	0.25	48	14	14.3	59
87	43.50	4.02	-0.15	4	4	19.8	6

I specify Mean Squared Error (MSE) to act as the loss function and use callbacks with the patience of 10 epoch values to prevent over-fitting. This gives me a total of $4,865$ trainable parameters. The data was then split into subsets train 70% and test 30%. Next, the target column values, ie. `domain1_score` was removed from the training set and passed to the above described sequential model. A validation set of 20% was reserved for tuning the model parameters. A sample history for 5 epochs is shown in *Table III*.

TABLE III
SEQUENTIAL MODEL TRAINING FOR CONTINUOUS VALUES

loss	mse	val_loss	val_mse	epoch
7.126871	7.126870	15.345133	15.345136	24
7.458091	7.458093	12.741349	12.741346	25
7.251735	7.251735	7.594893	7.594892	26
7.021521	7.021521	9.922328	9.922329	27
7.206810	7.206808	6.553464	6.553464	28

The experiment was run for all possible $8{+}1$ (`essay_set`) feature combinations. This resulted in 511 runs of the training process. For each iteration, predicted results are compared with the test set and plots are made like that in *Fig. 1*.

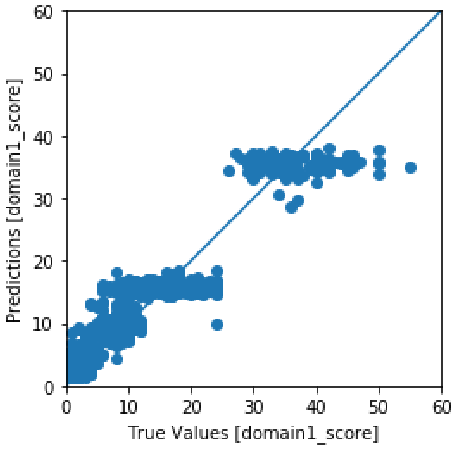

Fig. 1. Predicted Resolved Scores Vs. True Resolved Scores

IV. EVALUATION

The Quadratic Weighted Kappa is a widely used evaluation metric for inter-rater agreement and is the recommended evaluation mechanism by the Hewlett Foundation for their ASAP competition dataset. The Kappa score can range from -1 to 1. A negative, zero or low QWK indicates that the two rates only have a "chance" agreement while a positive score indicates that both the raters assign the same value in most cases. This is generally used for different human annotators but in this case, I compare the final machine predicted values (after round off) against the actual values given by human raters. The `sklearn` package provides an inbuilt function to compute QWK[7]. QWK scores above 0.8 are generally considered good.

When training across all features (ie. `'essay_set'`, `'essay_length'`, `'avg_sentence_length'`, `'avg_word_length'`, `'sentiment'`, `'noun_phrases'`, `'syntax_errors'`, `'readability_index'`, `'difficult_words'`), the model was able to achieve QWK score of 0.96. Below are a few more observations from the 511 runs that I found interesting:

- **Feature sets without `'essay_set'` in them performed poorly.** The inclusion of `'essay_set'` improved the overall QWK of the model in all cases. I am not sure about the reason for this behavior.

[7] https://scikit-learn.org/stable/modules/model_evaluation.html#cohen-kappa

- Ignoring `'essay_set'` from our superset of features, the best QWK was of 0.24 for `'essay_length'`, `'avg_sentence_length'`, `'avg_word_length'`, `'sentiment'`, `'noun_phrases'`, `'syntax_errors'`, `'readability_index'`, `'difficult_words'`.
- In case of **single feature sets, `'essay_length'` performed the best** when compared to others, achieving a QWK of 15. This makes sense since generally, a longer essay has a higher probability of being good and receiving a higher score when compared to a short one.
- In case of **single feature sets, `'sentiment'` performed the worst**, achieving a QWK of -0.00016. It was the only model to give negative or ”chance” agreement with the true values and perform worse than the baseline (random predictions). This too makes sense since ideally a grader would not be biased toward a negative or positive essay and hence score them neutrally. This may be the reason that sentiment alone did not prove to be a good feature for model training.

Most importantly, increasing the number of features[8] for model training did not always improve the QWK. This indicates that not only quantity, but quality of input features is also important. In certain cases, more features degraded the model to produce worse results than they did with a smaller subset.

V. Conclusion

I gained valuable experience working with several NLP packages and the TensorFlow platform using which I was able to implement a rudimentary AES system. The evaluation of the results indicates that the features selected were not good enough to deliver decent results. Further, there might be some logical flaws in the implementation which could be leading to high QWK on the inclusion of essay set features. Several improvements can be made to the current approach like extracting more prompt specific features, POS tag, etc. There is also a need for more extensive data cleaning to handle anonymized named entities so that they don’t contribute towards the count of syntax errors feature.

References

[1] Barseghian, Tina. “Can Robots Grade Essays As Well as Humans?,” February 14, 2012. https://www.kqed.org/mindshift/19019/can-robots-grade-essays-as-well-as-humans.

[2] “Hewlett Foundation Sponsors Prize to Improve Automated Scoring of Student Essays.” Hewlett Foundation, January 9, 2012. https://hewlett.org/newsroom/hewlett-foundation-sponsors-prize-to-improve-automated-scoring-of-student-essays/.

[3] ”How Long Does It Take To Grade An Essay?”. 2019. Teacherblue.Homestead.Com. http://teacherblue.homestead.com/gradingtime.html.

[4] “Test and Score Data”. 2019. Ets.Org. https://www.ets.org/s/toefl/pdf/94227_unlweb.pdf.

[5] Zhang, Mo. ”Contrasting automated and human scoring of essays.” R & D Connections 21, no. 2 (2013): 1-11.

[6] Attali, Yigal, and Jill Burstein. ”Automated essay scoring with e-rater® v. 2.0.” ETS Research Report Series 2004, no. 2 (2004): i-21.

[7] Zhao, Siyuan, Yaqiong Zhang, Xiaolu Xiong, Anthony Botelho, and Neil Heffernan. 2017. ”A Memory-Augmented Neural Model For Automated Grading”. Proceedings Of The Fourth (2017) ACM Conference On Learning @ Scale - L@S ’17. doi:10.1145/3051457.3053982.

[8] Yi, Bong-Jun, Do-Gil Lee, and Hae-Chang Rim. ”The effects of feature optimization on high-dimensional essay data.” Mathematical Problems in Engineering 2015 (2015).

[9] Attali, Yigal, and Jill Burstein. ”Automated essay scoring with e-rater® V. 2.” The Journal of Technology, Learning and Assessment 4, no. 3 (2006).

[10] Williams, Robert. ”Automated essay grading: An evaluation of four conceptual models.” In New horizons in university teaching and learning: Responding to change, pp. 173-184. Centre for Educational Advancement, Curtin University, 2001.

[11] Liang, Hong, Xiao Sun, Yunlei Sun, and Yuan Gao. ”Text feature extraction based on deep learning: a review.” EURASIP journal on wireless communications and networking 2017, no. 1 (2017): 1-12.

[12] Senter, R. J., and Edgar A. Smith. Automated readability index. CINCINNATI UNIV OH, 1967.

[8] example runs are available at https://asing80.people.uic.edu/cs421/results_df.tsv