# Automated Essay Scoring with Cross Feature Vector Generation Technique

Deshmukh Anup[1] and Patel Smit[2]

[1] International Institute of IT - Bangalore
IMT2014013
[2] International Institute of IT - Bangalore
IMT2014051

**Abstract.** It is becoming increasingly important to automate the correction of submitted answers. Verification of student answers is cumbersome and costly because human testers will have to manually assign marks to submitted solutions by comparing them with the correct solution. We call this correct solution as the golden standard. There exist some state of art techniques [6], [7] which simply check for complete correctness of submitted text answers against a set of test cases. It is equally important to find the distance of how far are these submitted text answers from the golden standard. Hence, we developed an automated approach to this problem using our model which has novel approach of feature vector generation of the text.

**Keywords:** *Automated Essay Grading, NLP, Regression*

## 1 Introduction

One of the difficulties of grading essays lies in the varying subjectivity of the grading process. This also leads to the contradicting variation in grades awarded by different human assessors, which is clearly unfair to the students. This problem could be solved by the adoption of automated assessment tools for essays(text answers). Lets have a look at the current approaches to the automated assessment of text answers.

### 1.1 Existing approaches for automatic essay grading

*Electronic Essay Rater (E-Rater):* E-Rater uses a combination of statistical and NLP techniques to extract linguistic features from the essays to be graded. [1]

*Conceptual Rater (C-Rater):* E-Rater assigns a score for writing skills rather than for specific-content while C-rater is aimed to score a response as being either correct or incorrect. [5]

*Intelligent Essay Assessor (IEA):* IEA was developed by (Hearst, 2000; Jerrams-Smith, Soh, Callear, 2001) and is based on the Latent Semantic Analysis (LSA) technique. [2]

*Based on text categorization:* The idea of automated essay grading based on text categorization techniques, text complexity features and linear regression methods was first explored by Larkey (1998) [4]. The underlying idea of this approach is to train the binary classifiers to distinguish good from bad essays and then using the scores produced by the classifiers to rank essays and assign grades to them.

While tackling this problem of essay grading, we emphasized on the following practical problem. It is very crucial that we reduce the size of data required for training our model as much as possible. This is because otherwise, a large proportion of the answers would require manual evaluation and hence the utility of the algorithm may be criticized. So our focus was on designing a model which is capable of producing good results on test data even after training it with a significantly small dataset.

## 2 Our Model: Intelligent Text Rater (ITR)

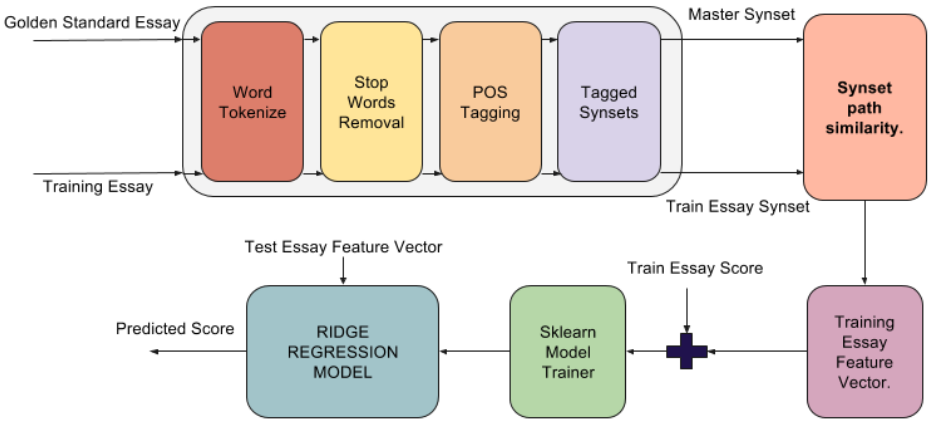


**Fig. 1.** Work flow of ITR

As depicted in the above flow-chart, the general work flow of our ITR is as follows:

- First up, the Reference Essay as supplied by the evaluator, undergoes word tokenization, stop words removal, Part-Of-Speech tagging and Synset-ID tagging in the same order to produce the set which we call as the *Master Synset.*
- The same procedure is used to generate a Synset for each one of essays in the training set. This Synset fundamentally is the set of sets wherein each set is of the form,

$$s_j = \text{('House', V, 1.0)}$$
$$Syn_i = (s_1, s_2, .. , s_{n_1})$$

where $n_1$ is the cardinality of $Syn_i$ and i corresponds to the $i^{th}$ training essay.

- Each training Synset from the list is then compared with the Master Synset using the Synset path-similarity algorithm and a corresponding feature vector for that training essay is produced. Details are explained in the section 2.1.
- Each feature vector from the list of training essay feature vectors is then combined with its corresponding score, as graded by the evaluator. They are then supplied to the Ridge Regression Model trainer with normalized coefficient equal to 0.5.
- This model then takes a test essay feature vector, produced from a test essay via the same pipeline as described above in Fig. 1, and score for that test essay is predicted.

### 2.1 Path similarity and Feature Vector Generation

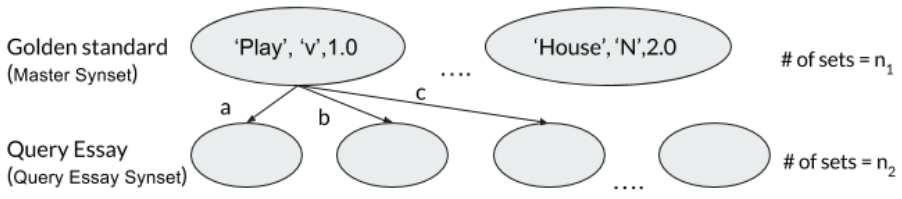


**Fig. 2.** Feature vector generation

This technique plays an vital role in decreasing the required training data size. Instead of directly generating feature vector from Synsets, we also incorporate the similarity between essays and the golden standard. The path similarity measure used by us returns the score denoting how similar two word senses are, based on the shortest path that connects the senses in the is-a (hypernym/hypnoym) taxonomy. The score is in the range 0 to 1. There are various path similarity measures which can be used to find the similarity between two Synsets. LCH similarity is based on the shortest path that connects the senses (as above) and the maximum depth of the taxonomy in which the senses occur. Wu-palmer similarity, which is based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer (most specific ancestor node). The procedure of Feature Vector generation can be described as follows:

- Each set present in the Master Synset is compared to every other set of the Query Synset, produced by processing the Query Essay. This similarity is

based on the path similarity algorithm of the NLTK Corpus Python library and is in a list of list format.
- For each list of length $n_2$ (from one iteration of above mentioned comparison), the max value is determined and stored. This max value is taken as one feature of the query essay.
- The above step is repeated for all the sets in Master Synset i.e $n_1$ number of times in Fig. 2
- This will produce a list of best-scores whose length is same as that of the our Master Synset.
- The list so produced is then used as a feature vector for our Ridge Regression Model Trainer.

## 3 Experiments

### 3.1 Dataset

Our dataset was divided in several sets of essays. We chose the first set which had a total of 1,784 essays. 70% of which was used for training while the rest of it for testing purpose. Each entry in the dataset contains an essay-set number, ASCII formatted exclusively textual essay, followed by a human score indicating the similarity of the essay with the domain. On average, each essay in the chosen set was approximately 150 to 550 words long and consisted of alphabets, numbers as well as special characters.[3]

*Hardware description:* All experiments were run on a standard Intel - Xeon 3.5GHz Quad-Core Processor and Corsair - Vengeance 2 x 16GB Memory

### 3.2 Evaluation

We used the mean squared error(MSE) as our evaluation metric. We tested our model on 50 randomly selected essays from 516 essays repeatedly, and the average of MSE score was noted.

**Table 1.** Number of training essays and MSE

| # training essays | MSE |
|---|---|
| 30 with RR($\alpha = 0.5$) | 0.729 |
| 20 with RR($\alpha = 0.5$) | 1.499 |
| 10 with RR($\alpha = 0.5$) | 1.99 |

### 3.3 Libraries used

| | |
|---|---|
| *NLTK* | For Word Tokenization and POS tagging. |
| *Sklearnl* | For implementing Ridge Regression Model. |
| *Pandas, Numpy and Regex* | For data manipulation. |

## 4 Concluding Remarks

In this paper, we explored supervised technique for automatic short answer grading. We believe our work made one important contribution. With the cross feature vector generation we were able to significantly reduce the training data size. We got the significantly good result even after training our model on 30 essays, each essay on an average being 300 words long. In future we intend to expand our work with the experimentations on different path similarity measures. We would also like to make our model more efficient both in terms of memory and time.

## References


1. Y. Attali and J. Burstein. Automated essay scoring with e-rater® v. 2.0. *ETS Research Report Series*, 2004(2), 2004.
2. P. W. Foltz, D. Laham, and T. K. Landauer. The intelligent essay assessor: Applications to educational technology. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 1(2):939–944, 1999.
3. https://www.kaggle.com/c/asap aes. The hewlett foundation: Automated essay scoring. HP, 2011.
4. L. S. Larkey. Automatic essay grading using text categorization techniques. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 90–95. ACM, 1998.
5. C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37(4):389–405, 2003.
6. M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 567–575. Association for Computational Linguistics, 2009.
7. S. Valenti, F. Neri, and A. Cucchiarelli. An overview of current research on automated essay grading. *Journal of Information Technology Education: Research*, 2(1):319–330, 2003.