Abstract

When creating a student response analysis (SRA) tool, one of the best approaches is to compare the similarity of a student’s answer to pre-tagged answers. Our system first spell checks and stems each student response. After this, it uses similarity metrics to build a logistic regression classifier for each question from our training set of questions and student answers. After the classifiers are built, they can then be used to classify the correctness of students’ answers. We tested this method against other classifiers and similarity metrics and found that it outperformed the baseline model in all cases.

1 Introduction

As technology advances, the way material is being taught is slowly shifting to a web-based paradigm. Massive open online course platforms such as Udacity, Coursera and UReddit are becoming popular tools for teaching what universities have been teaching for years. Currently, content mastery in these courses is typically tested using automated multiple choice grading. One shortfall with multiple choice questions is that students can guess and still get questions correct. Another issue is that students can pick up context clues and remove answers based on simple strategies, increasing the chance of a correct answer without any need for mastery of the topic being tested. A great solution for judging mastery is short-answer questions, but this requires instructor time to manually grade them. With class sizes no longer physically limited, manual grading is untenable.

In the field of written student tests, if there were a way to algorithmically pre-label student answers with a high degree of accuracy, it would greatly reduce the amount of time a teacher has to spend on grading. Additionally, such an automated system could be applied in other fields where there are unknown responses and classification of those responses is needed. The problem is that a student’s response needs to be interpreted, understood, and then categorized from that understanding.

The questions that our system works with cannot be open ended. As such, if the question relies on a closed domain of knowledge, we can look for keywords and their similar synonyms to help judge the correctness of an answer for a given question.

For our system, a sample of correct answers for questions in domains with specific keywords was provided. Armed with a set of responses it knows are correct, our system has a set of knowledge before it begins testing on student responses. If it can associate certain keywords with correctness, incorrectness, partial correctness, or irrelevancy, it can then evaluate a student’s response. If the student has a response similar to a certain label, it can be labelled as such.

One research problem is the question of how to define words as similar. Much research has gone into this topic and we found many tools which could be applied to system. We checked libraries from OpenNLP, Stanford, Weka, and similarity wordnets. In the end, we experimented with all of them and found a similarity wordnet to be the most effective.

2 Related Work

2.1 C-Rater

C-Rater is an automated scorer for short answers such as those our system evaluates. Unlike previous research which focused on essay structure, C-Rater focused on the words that were used in the student’s response. Where our system differs is the additional flags that C-Rater focuses on. It also adds predicate argument structure, pronounal reference, and morphological analysis. One
concern with C-Rater was that it focuses mostly on ensuring the short answers are also grammatically correct, in addition to checking the correctness of their content. As such, we focused less on these additions, since the SRA task is only concerned with whether a response is within a certain threshold of correctness, not on how well the student can write out the answer. Our system is even more lenient in that it includes a spelling correction and stemming phase before testing similarity.

2.2 CarmelTC

CarmelTC was an approach to classify student essays using a hybrid system that combined syntactic analysis with Naïve Bayes classification and the “bag of words” similarity metric. Our system’s domain differs significantly, but its approach is similar. In both systems, syntactic analysis is done, a classifier is trained, and similarity metrics are run. Our system does not use Naïve Bayes, since the SRA task is categorical. For our similarity metric, we use a networked approach instead of “bag of words”, since short responses have less words.

2.3 Entailment

The developers of the tutoring system iSTART produced a system that focused entirely on the entailment of text, i.e. whether or not a text was the consequence of some previous text or texts. Though our system does not perform a textual entailment analysis, the researchers showed that syntactic measures paired with entailment notably improved their system’s accuracy. In keeping with this idea, our system’s first step was to correct syntactic errors in the student’s response, in order to increase classification accuracy.

3 Design Methodology

3.1 Classifiers

To guide the design of our system, we tested both local (per-question) and global (per-corpus) classifiers using several different classifiers.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Local</th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td>RandomForest</td>
<td>94%</td>
<td>57%</td>
</tr>
<tr>
<td>Multilayer Perceptron</td>
<td>68%</td>
<td>47%</td>
</tr>
<tr>
<td>SMO</td>
<td>58%</td>
<td>46%</td>
</tr>
</tbody>
</table>

Table 1: Accuracy of local vs. global classifiers.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>beetle</th>
<th>seb</th>
</tr>
</thead>
<tbody>
<tr>
<td>RandomForest</td>
<td>95.62%</td>
<td>97.06%</td>
</tr>
<tr>
<td>Logistic</td>
<td>83.61%</td>
<td>78.65%</td>
</tr>
<tr>
<td>SMO</td>
<td>69.44%</td>
<td>68.09%</td>
</tr>
</tbody>
</table>

Table 2: Training results with various classifiers, compared to the baseline model.

Due to the substantial effect the choice of classifier has on system accuracy, we tested RandomForest, Multilayer Perceptron, and SMO, with both local and global classifiers. To evaluate these classifiers, our system was both trained and tested on the beetle training set, which was expected to overfit the data and provide high accuracy. As shown in Table 1, local classifiers give more accurate results across the board.

3.1.2 Choosing the Right Classifier

After deciding to use a local (per-question) classification scheme, the design focus shifted to choosing a classifier from the many classifiers provided by the Weka library. We explored three options: RandomForest, Logistic (a logistic regression model), and SMO (a support vector machine implementation).

The system’s goal is to produce the most accurate labels for unknown student responses. In most of the preliminary tests, the training set was also the test set, which can cause inaccurate results since the classifier might overfit the test data to the training data.

Based on the results in Table 2, while RandomForest produces the highest accuracy in this setting, declaring it the best choice may be
Similarity

<table>
<thead>
<tr>
<th>Library</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenNLP – Similarity</td>
<td>0.85</td>
</tr>
<tr>
<td>Weka – Euclidean Distance</td>
<td>0.88</td>
</tr>
<tr>
<td>Levenshtein Distance</td>
<td>0.74</td>
</tr>
<tr>
<td>YTEX</td>
<td>0.59</td>
</tr>
<tr>
<td>Vector-space EOWL metric</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 3: Average test sentence similarity for various similarity metrics.

specious, as it has a tendency to overfit on small datasets. This effect is magnified on datasets with low-order feature vectors, such as those produced by local classifiers. Since logistic regression still provides great accuracy but has less of an overfitting problem, we selected Logistic as the basis for our classifier.

3.2 Semantic Similarity

We tested the similarity metrics of five different libraries in order to obtain the best for our system. In order to test for similarity, five sample sentences were selected from the seb corpus. Common words were manually replaced with synonyms and then each sentence was compared with its original using each library, in turn. A perfect similarity metric, then, should return a similarity of 1.0 for all test cases.

Based on Table 3, the best performance was achieved using a vector-space EOWL metric. Thus, this was chosen for the semantic similarity phase of our system’s pipeline.

3.3 Syntactic Analysis

3.3.1 Spell Checking and Stop Words

In order to further improve our system’s similarity metric, it first spell checks student responses. If the edit distance is below a threshold, the system automatically corrects the words.

Some words, such as “a” and “the”, may be found to contribute to similarity when they should not. Since these words contribute little meaning, a list of stop words was created so as to remove any occurrence of such words from both student responses and tagged responses.

3.3.2 Stemming

The next phase of our system’s syntactic analysis is stemming calculations. Since, for example, words such as “swimming” and “swim” have the same stem, our system isolates the stem since affixes could potentially lower computed similarity if the wordnet does not contain adjacencies for stemmed words.

3.3.3 SimMetrics

The final phase of the syntactic analysis uses the SimMetrics package. Specifically, the Jaro–Winkler distance is used to calculate the overlap between two words. Jaro–Winkler was developed by the U.S. Census Bureau to automatically correlate closely transposed words, which makes it ideal for cases of simple typographical errors. We found Jaro–Winkler distance to be better than Levenshtein distance, increasing our system’s accuracy by 1 to 2%.

4 Experimental Results

Table 4 contains experimental results from running the test set against various classifier models. In all cases, our system performed better than the baseline model. Of note is that, when run on unknown student responses, RandomForest still tended to perform as well as or better than Logistic.

5 Analysis

RandomForest outperforms logistic regression by a substantial margin on the beetle test set. Although this would seem to go against our theory that RandomForest was overfitting its classifiers, it is our belief that this is because the test set for beetle is less varied than that of seb. The test set for seb is relatively small and thus conclusions drawn from its results must be weighed against such considerations. Further, Logistic outperforms RandomForest on seb only by a small margin.

Overall, both algorithms are likely to result in comparable performance. Fine tuning of algorithm parameters is likely to yield further performance gains. Whether RandomForest or Logistic is superior for this task remains an open question. It is our belief that
RandomForest is likely to yield decent performance in the average case. For instances where the test set is likely to be substantially different than the train set, Logistic may provide better results. More broadly, logistic regression and random forest question-level classification both outperform the baseline model in both test sets, though classifier choice still has substantial impact on overall system performance.

6 Discussion

It may be argued that the use of per-question classifiers is impractical for a real-world SRA system. It is our belief that this concern is minimal for most applications. For instance, in a tutoring dialogue system or an automated essay-grading system, the system should have all questions and reference answers a priori and will have had the opportunity to train beforehand, dynamically loading the appropriate models at runtime. As new questions are added to the system, new classifiers could then be trained and added automatically.

Despite this, being able to handle unseen questions and answers may still be useful. Our approach in this regard is simple and elegant. For each model the system creates, an average value for each class (correct, incorrect, etc.) is stored into a feature vector. After all models are created, a $k$-means clustering algorithm is trained on this dataset. At runtime, when an unseen question is encountered, its reference answers are used to construct a vector corresponding to the average values for each class. This vector is then submitted to the $k$-means classifier, resulting in a reference to an existing question model which best handles the new inputs.

We feel that such an approach will adequately address the vast majority of situations. Optionally, this process could be instrumented with a check of the $k$-means confidence score for the prediction. In the case that the confidence score was below a certain threshold, the system would simply back-off to a global classifier.

One valid concern with this design is that because each classifier trains on so few student answers, not all possible question tags may be covered. Extreme cases could arise where a classifier is only trained on data from one classification and would always return the class it saw during training. We did not handle such cases in our experimental setup. We computed a per-question error rate and did not encounter such anomalous cases; however, this does not preclude their existence. A simple approach to handle such cases is to borrow training data from other questions whose student and tagged responses have similar semantics to the sparse question reference answers.

7 Conclusion

In this study, we developed a system for student response analysis (SRA), which utilized semantic and syntactic similarity metrics. Local and global classifiers were tested for accuracy and local classifiers were employed after cross-validation analysis determined their accuracy to outperform global strategies. Various machine learning algorithms were tested; however, due to concerns over overfitting bias, Logistic was chosen over RandomForest. Ultimately, RandomForest and Logistic give comparable performance on the test set, though RandomForest tends to yield superior performance on test sets more similar to the training data. The results of this study yield insight into classifier choice in SRA system design and suggest new directions in textual similarity research.

References


