Proceedings of the 51st Annual Meeting of the Association for
Computational Linguistics (ACL), Sofia, Bulgaria, August 2013, pp. 260--269.

Modeling Thesis Clarity in Student Essays

Isaac Persing and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{persingq,vince}@hlt.utdallas.edu

Abstract

Recently, researchers have begun exploring methods of scoring student essays with respect to particular dimensions of quality such as coherence, technical errors, and relevance to prompt, but there is relatively little work on modeling thesis clarity. We present a new annotated corpus and propose a learning-based approach to scoring essays along the thesis clarity dimension. Additionally, in order to provide more valuable feedback on why an essay is scored as it is, we propose a second learning-based approach to identifying what kinds of errors an essay has that may lower its thesis clarity score.

1 Introduction

Automated essay scoring, the task of employing computer technology to evaluate and score written text, is one of the most important educational applications of natural language processing (NLP) (see Shermis and Burstein (2003) and Shermis et al. (2010) for an overview of the state of the art in this task). A major weakness of many existing scoring engines such as the Intelligent Essay Assessor$^{\text{TM}}$(Landauer et al., 2003) is that they adopt a holistic scoring scheme, which summarizes the quality of an essay with a single score and thus provides very limited feedback to the writer. In particular, it is not clear which dimension of an essay (e.g., style, coherence, relevance) a score should be attributed to. Recent work addresses this problem by scoring a particular dimension of essay quality such as coherence (Miltsakaki and Kukich, 2004), technical errors, relevance to prompt (Higgins et al., 2004), and organization (Persing et al., 2010). Essay grading software that provides feedback along multiple dimensions of essay quality such as E-*rater*/Criterion (Attali and Burstein, 2006) has also begun to emerge.

Nevertheless, there is an essay scoring dimension for which few computational models have been developed — *thesis clarity*. Thesis clarity refers to how clearly an author explains the *thesis* of her essay, i.e., the position she argues for with respect to the topic on which the essay is written.[1] An essay with a high thesis clarity score presents its thesis in a way that is easy for the reader to understand, preferably but not necessarily directly, as in essays with explicit thesis sentences. It additionally contains no errors such as excessive misspellings that make it more difficult for the reader to understand the writer's purpose.

Our goals in this paper are two-fold. First, we aim to develop a computational model for scoring the thesis clarity of student essays. Because there are many reasons why an essay may receive a low thesis clarity score, our second goal is to build a system for determining why an essay receives its score. We believe the feedback provided by this system will be more informative to a student than would a thesis clarity score alone, as it will help her understand which aspects of her writing need to be improved in order to better convey her thesis. To this end, we identify five common errors that impact thesis clarity, and our system's purpose is to determine which of these errors occur in a given essay. We evaluate our thesis clarity scoring model and error identification system on a data set of 830 essays annotated with both thesis clarity scores and errors.

In sum, our contributions in this paper are three-fold. First, we develop a scoring model and error identification system for the thesis clarity dimension on student essays. Second, we use features explicitly designed for each of the identified error

[1] An essay's thesis is the overall message of the *entire* essay. This concept is unbound from the the concept of thesis sentences, as even an essay that never explicitly states its thesis in any of its sentences may still have an overall message that can be inferred from the arguments it makes.

Topic	Languages	Essays
Most university degrees are theoretical and do not prepare students for the real world. They are therefore of very little value.	13	131
The prison system is outdated. No civilized society should punish its criminals: it should rehabilitate them.	11	80
In his novel *Animal Farm*, George Orwell wrote "All men are equal but some are more equal than others." How true is this today?	10	64

Table 1: Some examples of writing topics.

types in order to train our scoring model, in contrast to many existing systems for other scoring dimensions, which use more general features developed without the concept of error classes. Third, we make our data set consisting of thesis clarity annotations of 830 essays publicly available in order to stimulate further research on this task. Since progress in thesis clarity modeling is hindered in part by the lack of a publicly annotated corpus, we believe that our data set will be a valuable resource to the NLP community.

2 Corpus Information

We use as our corpus the 4.5 million word International Corpus of Learner English (ICLE) (Granger et al., 2009), which consists of more than 6000 essays written by university undergraduates from 16 countries and 16 native languages who are learners of English as a Foreign Language. 91% of the ICLE texts are argumentative. We select a subset consisting of 830 argumentative essays from the ICLE to annotate and use for training and testing of our models of essay thesis clarity. Table 1 shows three of the thirteen topics selected for annotation. Fifteen native languages are represented in the set of essays selected for annotation.

3 Corpus Annotation

For each of the 830 argumentative essays, we ask two native English speakers to (1) score it along the thesis clarity dimension and (2) determine the subset of the five pre-defined errors that detracts from the clarity of its thesis.

Scoring. Annotators evaluate the clarity of each essay's thesis using a numerical score from 1 to 4 at half-point increments (see Table 2 for a description of each score). This contrasts with previous work on essay scoring, where the corpus is annotated with a binary decision (i.e., *good* or *bad*) for a given scoring dimension (e.g., Higgins et al. (2004)). Hence, our annotation scheme not only provides a finer-grained distinction of thesis clarity (which can be important in practice), but also makes the prediction task more challenging.

Score	Description of Thesis Clarity
4	essay presents a **very clear thesis** and requires little or no clarification
3	essay presents a **moderately clear thesis** but could benefit from some clarification
2	essay presents an **unclear thesis** and would greatly benefit from further clarification
1	essay presents **no thesis of any kind** and it is difficult to see what the thesis could be

Table 2: Descriptions of the meaning of scores.

To ensure consistency in annotation, we randomly select 100 essays to have graded by both annotators. Analysis of these essays reveals that, though annotators only exactly agree on the thesis clarity score of an essay 36% of the time, the scores they apply are within 0.5 points in 62% of essays and within 1.0 point in 85% of essays. Table 3 shows the number of essays that receive each of the seven scores for thesis clarity.

score	1.0	1.5	2.0	2.5	3.0	3.5	4.0
essays	4	9	52	78	168	202	317

Table 3: Distribution of thesis clarity scores.

Error identification. To identify what kinds of errors make an essay's thesis unclear, we ask one of our annotators to write 1–4 sentence critiques of thesis clarity on 527 essays, and obtain our list of five common error classes by categorizing the things he found to criticize. We present our annotators with descriptions of these five error classes (see Table 4), and ask them to assign zero or more of the error types to each essay.

It is important to note that we ask our annotators to mark an essay with one of these errors only when the error makes the thesis less clear. So for example, an essay whose thesis is irrelevant to the prompt but is explicitly and otherwise clearly stated would not be marked as having a Relevance to Prompt error. If the irrelevant thesis is stated in such a way that its inapplicability to the prompt causes the reader to be confused about what the essay's purpose is, however, then the essay would be assigned a Relevance to Prompt error.

To measure inter-annotator agreement on error identification, we ask both annotators to identify

Id	Error	Description
CP	**Confusing Phrasing**	The thesis is phrased oddly, making it hard to understand the writer's point.
IPR	**Incomplete Prompt Response**	The thesis seems to leave some part of a multi-part prompt unaddressed.
R	**Relevance to Prompt**	The apparent thesis's weak relation to the prompt causes confusion.
MD	**Missing Details**	The thesis leaves out important detail needed to understand the writer's point.
WP	**Writer Position**	The thesis describes a position on the topic without making it clear that this is the position the writer supports.

Table 4: Descriptions of thesis clarity errors.

the errors in the same 100 essays that were doubly-annotated with thesis clarity scores. We then compute Cohen's Kappa (Carletta, 1996) on each error from the two sets of annotations, obtaining an average Kappa value of 0.75, which indicates fair agreement. Table 5 shows the number of essays assigned to each of the five thesis clarity errors. As we can see, Confusing Phrasing, Incomplete Prompt Response, and Relevance to Prompt are the major error types.

error	CP	IPR	R	MD	WP
essays	152	123	142	47	39

Table 5: Distribution of thesis clarity errors.

Relationship between clarity scores and error classes. To determine the relationship between thesis clarity scores and the five error classes, we train a linear SVM regressor using the SVM^{light} software package (Joachims, 1999) with the five error types as independent variables and the reduction in thesis clarity score due to errors as the dependent variable. More specifically, each training example consists of a target, which we set to the essay's thesis clarity score minus 4.0, and six binary features, each of the first five representing the presence or absence of one of the five errors in the essay, and the sixth being a bias feature which we always set to 1. Representing the reduction in an essay's thesis clarity score with its thesis clarity score minus 4.0 allows us to more easily interpret the error and bias weights of the trained system, as under this setup, each error's weight should be a negative number reflecting how many points an essay loses due to the presence of that error. The bias feature allows for the possibility that an essay may lose points from its thesis clarity score for problems not accounted for in our five error classes. By setting this bias feature to 1, we tell our learner that an essay's default score may be less than 4.0 because these other problems may lower the average score of otherwise perfect essays.

After training, we examined the weight parameters of the learned regressor and found that they were all negative: -0.6 for CP, -0.5998 for IPR, -0.8992 for R, -0.6 for MD, -0.8 for WP, and -0.1 for the bias. These results are consistent with our intuition that each of the enumerated error classes has a negative impact on thesis clarity score. In particular, each has a demonstrable negative impact, costing essays an average of more than 0.59 points when it occurs. Moreover, this set of errors accounts for a large majority of all errors impacting thesis clarity because unenumerated errors cost essays an average of only one-tenth of one point on the four-point thesis clarity scale.

4 Error Classification

In this section, we describe in detail our system for identifying thesis clarity errors.

4.1 Model Training and Application

We recast the problem of identifying which thesis clarity errors apply to an essay as a multi-label classification problem, wherein each essay may be assigned zero or more of the five pre-defined error types. To solve this problem, we train five binary classifiers, one for each error type, using a one-versus-all scheme. So in the binary classification problem for identifying error e_i, we create one training instance from each essay in the training set, labeling the instance as positive if the essay has e_i as one of its labels, and negative otherwise. Each instance is represented by seven types of features, including two types of baseline features (Section 4.2) and five types of features we introduce for error identification (Section 4.3).

After creating training instances for error e_i, we train a binary classifier, b_i, for identifying which test essays contain error e_i. We use SVM^{light} for classifier training with the regularization parameter, C, set to c_i. To improve classifier performance, we perform feature selection. While we employ seven types of features (see Sections 4.2 and 4.3), only the word n-gram features are subject to feature selection.[2] Specifically, we employ

[2] We do not apply feature selection to the remaining fea-

the top n_i n-gram features as selected according to information gain computed over the training data (see Yang and Pedersen (1997) for details). Finally, since each classifier assigns a real value to each test essay presented to it indicating its confidence that the essay should be assigned error e_i, we employ a classification threshold t_i to decide how high this real value must be in order for our system to conclude that an essay contains error e_i.

Using held-out validation data, we jointly tune the three parameters in the previous paragraph, c_i, n_i, and t_i, to optimize the F-score achieved by b_i for error e_i.[3] However, an exact solution to this optimization problem is computationally expensive. Consequently, we find a local maximum by employing the simulated annealing algorithm (Kirkpatrick et al., 1983), altering one parameter at a time to optimize F-score by holding the remaining parameters fixed.

After training the classifiers, we use them to classify the test set essays. The test instances are created in the same way as the training instances.

4.2 Baseline Features

Our **B**aseline system for error classification employs two types of features. First, since labeling essays with thesis clarity errors can be viewed as a text categorization task, we employ lemmatized word unigram, bigram, and trigram features that occur in the essay that have not been removed by the feature selection parameter n_i. Because the essays vary greatly in length, we normalize each essay's set of word features to unit length.

The second type of baseline features is based on random indexing (Kanerva et al., 2000). Random indexing is "an efficient, scalable and incremental alternative" (Sahlgren, 2005) to Latent Semantic Indexing (Deerwester et al., 1990; Landauer and Dutnais, 1997) which allows us to automatically generate a semantic similarity measure between any two words. We train our random indexing model on over 30 million words of the English Gigaword corpus (Parker et al., 2009) using the S-Space package (Jurgens and Stevens, 2010). We expect that features based on random indexing may be particularly useful for the Incomplete Prompt Response and Relevance to Prompt errors because they may help us find text related to the prompt even if some of its components have been rephrased (e.g., an essay may talk about "jail" rather than "prison", which is mentioned in one of the prompts). For each essay, we therefore generate four random indexing features, one encoding the entire essay's similarity to the prompt, another encoding the essay's highest individual sentence's similarity to the prompt, a third encoding the highest entire essay similarity to one of the prompt sentences, and finally one encoding the highest individual sentence similarity to an individual prompt sentence. Since random indexing does not provide a straightforward way to measure similarity between groups of words such as sentences or essays, we use Higgins and Burstein's (2007) method to generate these features.

4.3 Novel Features

Next, we introduce five types of novel features.

Spelling. One problem we note when examining the information gain top-ranked features for the Confusing Phrasing error is that there are very few common confusing phrases that can contribute to this error. Errors of this type tend to be unique, and hence are not very useful for error classification (because we are not likely to see the same error in the training and test sets). We notice, however, that there are a few misspelled words at the top of the list. This makes sense because a thesis sentence containing excessive misspellings may be less clear to the reader. Even the most common spelling errors, however, tend to be rare. Furthermore, we ask our annotators to only annotate an error if it makes the thesis less clear. The mere presence of an awkward phrase or misspelling is not enough to justify the Confusing Phrasing label. Hence, we introduce a **m**isspelling feature whose value is the number of spelling errors in an essay's most-misspelled sentence.[4]

ture types since each of them includes only a small number of overall features that are expected to be useful.

[3] For parameter tuning, we employ the following values. c_i may be assigned any of the values 10^2, 10^3, 10^4, 10^5, or 10^6. n_i may be assigned any of the values 3000, 4000, 5000, or ALL, where ALL means all features are used. For t_i, we split the range of classification values b_i returns for the test set into tenths. t_i may take the values 0.0, 0.1, 0.2, . . ., 1.0, and X, where 0.0 classifies all instances as negative, 0.1 classifies only instances b_i assigned values in the top tenth of the range as positive, and so on, and X is the default threshold, labeling essays as positive instances of e_i only if b_i returns for them a value greater than 0. It was necessary to assign t_i in this way because the range of values classifiers return varies greatly depending on which error type we are classifying and which other parameters we use. This method gives us reasonably fine-grained thresholds without having to try an unreasonably large number of values for t_i.

[4] We employ SCOWL (`http://wordlist.sourceforge.net/`) as our dictionary, assuming that a

Keywords. Improving the prediction of majority classes can greatly enhance our system's overall performance. Hence, since we have introduced the misspelling feature to enhance our system's performance on one of the more frequently occurring errors (Confusing Phrasing), it makes sense to introduce another type of feature to improve performance on the other two most frequent errors, Incomplete Prompt Response and Relevance to Prompt. For this reason, we introduce **k**eyword features. To use this feature, we first examine each of the 13 essay prompts, splitting it into its component pieces. For our purposes, a component of a prompt is a prompt substring such that, if an essay does not address it, it may be assigned the Incomplete Prompt Response label. Then, for each component, we manually select the most important (primary) and second most important (secondary) words that it would be good for a writer to use to address the component. To give an example, the lemmatized version of the third component of the second essay in Table 1 is "it should rehabilitate they". For this component we selected "rehabilitate" as a primary keyword and "society" as a secondary keyword. To compute one of our keyword features, we compute the random indexing similarity between the essay and each group of primary keywords taken from components of the essay's prompt and assign the feature the lowest of these values. If this feature has a low value, that suggests that the essay may have an Incomplete Prompt Response error because the essay probably did not respond to the part of the prompt from which this value came. To compute another of the keyword features, we count the numbers of combined primary and secondary keywords the essay contains from each component of its prompt, and divide each number by the total number of primary and secondary features for that component. If the greatest of these fractions has a low value, that indicates the essay's thesis might not be very Relevant to the Prompt.[5]

Aggregated word n-grams. Other ways we could measure our system's performance (such as macro F-score) would consider our system's performance on the less frequent errors no less important than its performance on the most frequent errors. For this reason, it now makes sense for us to introduce a feature tailored to help our system do better at identifying the least-frequent error types, Missing Details and Writer Position, each of which occurs in fewer than 50 essays. To help with identification of these error classes, we introduce aggregated **w**ord n-gram features. While we mention in the previous section one of the reasons regular word n-gram features can be expected to help with these error classes, one of the problems with regular word n-gram features is that it is fairly infrequent for the exact same useful phrase to occur too frequently. Additionally, since there are numerous word n-grams, some infrequent ones may just by chance only occur in positive training set instances, causing the learner to think they indicate the positive class when they do not. To address these problems, for each of the five error classes e_i, we construct two Aggregated word features $Aw+_i$ and $Aw-_i$. For each essay, $Aw+_i$ counts the number of word n-grams we believe indicate that an essay is a positive example of e_i, and $Aw-_i$ counts the number of word n-grams we believe indicate an essay is not an example of e_i. $Aw+$ n-grams for the Missing Details error tend to include phrases like "there is something" or "this statement", while $Aw-$ n-grams are often words taken directly from an essay's prompt. N-grams used for Writer Position's $Aw+$ tend to suggest the writer is distancing herself from whatever statement is being made such as "every person", but n-grams for this error's $Aw-$ feature are difficult to find. Since $Aw+_i$ and $Aw-_i$ are so error specific, they are only included in an essay's feature representation when it is presented to learner b_i. So while aggregated word n-grams introduce ten new features, each learner b_i only sees two of these ($Aw+_i$ and $Aw-_i$).

We construct the lists of word n-grams that are aggregated for use as the $Aw+$ and $Aw-$ feature values in the following way. For each error class e_i, we sort the list of all features occurring at least ten times in the training set by information gain. A human annotator then manually inspects the top thousand features in each of the five lists and sorts each list's features into three categories. The first category for e_i's list consists of features that indicate an essay may be a positive instance. Each word n-gram from this list that occurs in an essay increases the essay's $Aw+_i$ value by one. Similarly, any word n-gram sorted into the second

word that does not appear in the dictionary is misspelled.

[5] Space limitations preclude a complete listing of the keyword features. See our website at `http://www.hlt.utdallas.edu/~persingq/ICLE/` for the complete list.

category, which consists of features the annotator thinks indicate a negative instance of e_i, increases the essay's $Aw-$ value by one. The third category just contains all the features the annotator did not believe were useful enough to either class, and we make no further use of those features. For most error types, only about 12% of the top 1000 features get sorted into one of the first two categories.

Aggregated POS n-grams. We might further improve our system's performance on the Missing Details error type by introducing a feature that aggregates **p**art-of-speech (POS) tag n-grams in the same way that the Aw features aggregate word n-gram features. For this reason, we include POS tag 1, 2, 3, and 4-grams in the set of features we sort in the previous paragraph. For each error e_i, we select POS tag n-grams from the top thousand features of the information gain sorted list to count toward the $Ap+_i$ and $Ap-_i$ aggregation features. We believe this kind of feature may help improve performance on Missing Details because the list of features aggregated to generate the $Ap+_i$ feature's value includes POS n-gram features like CC " NN " (scare quotes). This feature type may also help with Confusing Phrasing because the list of POS tag n-grams our annotator generated for its $Ap+_i$ contains useful features like DT NNS VBZ VBN (e.g., "these signals has been"), which captures noun-verb disagreement.

Aggregated frames. Our last aggregated feature is generated using FrameNet-style semantic role labels obtained using SEMAFOR (Das et al., 2010). For each sentence in our data set, SEMAFOR identifies each semantic frame occurring in the sentence as well as each frame element that participates in it. For example, a semantic frame may describe an event that occurs in a sentence, and the event's frame elements may be the people or objects that participate in the event. For a more concrete example, consider the sentence "They said they do not believe that the prison system is outdated". This sentence contains a Statement frame because a statement is made in it. One of the frame elements participating in the frame is the Speaker "they". From this frame, we would extract a feature pairing the frame together with its frame element to get the feature "Statement-Speaker-they". This feature indicates that the essay it occurs in might be a positive instance of the Writer Position error since it tells us the writer is attributing some statement being made to someone else. Hence, this feature along with several others like "Awareness-Cognizer-we all" are useful when constructing the lists of frame features for Writer Position's aggregated **f**rame features $Af+_i$ and $Af-_i$. Like every other aggregated feature, $Af+_i$ and $Af-_i$ are generated for every error e_i.

5 Score Prediction

Because essays containing thesis clarity errors tend to have lower thesis clarity scores than essays with fewer errors, we believe that thesis clarity scores can be predicted for essays by utilizing the same features we use for identifying thesis clarity errors. Because our score prediction system uses the same feature types we use for thesis error identification, each essay's vector space representation remains unchanged. Only its label changes to one of the values in Table 2 in order to reflect its thesis clarity score. To make use of the fact that some pairs of scores are more similar than others (e.g., an essay with a score of 3.5 is more similar to an essay with a score of 4.0 than it is to one with a score of 1.0), we cast thesis clarity score prediction as a regression rather than classification task.

Treating thesis clarity score prediction as a regression problem removes our need for a classification threshold parameter like the one we use in the error identification problem, but if we use SVM^{light}'s regression option, it does not remove the need for tuning a regularization parameter, C, or a feature selection parameter, n.[6] We jointly tune these two parameters to optimize performance on held-out validation data by performing an exhaustive search in the parameter space.[7]

After we select the features, construct the essay instances, train a regressor on training set essays, and tune parameters on validation set essays, we can use the regressor to obtain thesis clarity scores on test set essays.

[6] Before tuning the feature selection parameter, we have to sort the list of n-gram features occurring the training set. To enable the use of information gain as the sorting criterion, we treat each distinct score as its own class.

[7] The absence of the classification threshold parameter and the fact that we do not need to train multiple learners, one for each score, make it feasible for us to do two things. First, we explore a wider range of values for the two parameters: we allow C to take any value from 10^0, 10^1, 10^2, 10^3, 10^4, 10^5, 10^6, or 10^7, and we allow n to take any value from 1000, 2000, 3000, 4000, 5000, or ALL. Second, we exhaustively explore the space defined by these parameters in order to obtain an exact solution to the parameter optimization problem.

6 Evaluation

In this section, we evaluate our systems for error identification and scoring. All the results we report are obtained via five-fold cross-validation experiments. In each experiment, we use 3/5 of our labeled essays for model training, another 1/5 for parameter tuning, and the final 1/5 for testing.

6.1 Error Identification

Evaluation metrics. To evaluate our thesis clarity error type identification system, we compute precision, recall, micro F-score, and macro F-score, which are calculated as follows. Let tp_i be the number of test essays correctly labeled as positive by error e_i's binary classifier b_i; p_i be the total number of test essays labeled as positive by b_i; and g_i be the total number of test essays that belong to e_i according to the gold standard. Then, the precision (P_i), recall (R_i), and F-score (F_i) for b_i and the macro F-score ($\hat{\mathrm{F}}$) of the combined system for one test fold are calculated by

$$\mathrm{P}_i = \frac{tp_i}{p_i}, \mathrm{R}_i = \frac{tp_i}{g_i},\ \mathrm{F}_i = \frac{2\mathrm{P}_i\mathrm{R}_i}{\mathrm{P}_i + \mathrm{R}_i}, \hat{\mathrm{F}} = \frac{\sum_i \mathrm{F}_i}{5}.$$

However, the macro F-score calculation can be seen as giving too much weight to the less frequent errors. To avoid this problem, we also calculate for each system the micro precision, recall, and F-score (P, R, and F), where

$$\mathrm{P} = \frac{\sum_i tp_i}{\sum_i p_i}, \mathrm{R} = \frac{\sum_i tp_i}{\sum_i g_i}, \mathrm{F} = \frac{2\mathrm{PR}}{\mathrm{P} + \mathrm{R}}.$$

Since we perform five-fold cross-validation, each value we report for each of these measures is an average over its values for the five folds.[8]

Results and discussion. Results on error identification, expressed in terms of precision, recall, micro F-score, and macro F-score are shown in the first four columns of Table 6. Our **B**aseline system, which only uses word n-gram and random indexing features, seems to perform uniformly poorly across both micro and macro F-scores (F and $\hat{\mathrm{F}}$; see row 1). The per-class results[9] show that, since micro F-score places more weight on the correct identification of the most frequent errors, the system's micro F-score (31.1%) is fairly close to the average of the scores obtained on the three most frequent error classes, CP, IPR, and R, and remains unaffected by very low F-scores on the two remaining infrequent classes.[10]

		Error Identification				Scoring		
	System	P	R	F	$\hat{\mathrm{F}}$	$S1$	$S2$	$S3$
1	**B**	24.8	44.7	31.1	24.0	.658	.517	.403
2	**Bm**	24.2	44.2	31.2	25.3	.654	.515	.402
3	**Bmk**	29.2	44.2	34.9	26.7	.663	.490	**.369**
4	**Bmkw**	28.5	49.6	35.5	31.4	**.651**	.484	.374
5	**Bmkwp**	**34.2**	49.6	40.4	34.6	.671	**.483**	.377
6	**Bmkwpf**	33.6	**54.4**	**41.4**	**37.3**	.672	.486	.382

Table 6: Five-fold cross-validation results for thesis clarity error identification and scoring.

When we add the **m**isspelling feature to the baseline, resulting in the system called **Bm** (row 2), the micro F-score sees a very small, insignificant improvement.[11] What is pleasantly surprising, however, is that, even though the misspelling features were developed for the Confusing Phrasing error type, they actually have more of a positive impact on Missing Details and Writer Position, bumping their individual error F-scores up by about 5 and 3 percent respectively. This suggests that spelling difficulties may be correlated with these other essay-writing difficulties, despite their apparent unrelatedness. This effect is strong enough to generate the small, though insignificant, gain in macro F-score shown in the table.

When we add **k**eyword features to the system, micro F-score increases significantly by 3.7 points (row 3). The micro per-class results reveal that, as intended, keyword features improve Incomplete Prompt Response and Relevance to Prompt's F-scores reveals that they do by 6.4 and 9.2 percentage points respectively. The macro F-scores reveal this too, though the macro F-score gains are 3.2 points and 11.5 points respectively. The macro F-score of the overall system would likely have improved more than shown in the table if the addition of keyword features did not simultaneously reduce Missing Details's score by several points.

While we hoped that adding aggregated **w**ord n-gram features to the system (row 4) would be able to improve performance on Confusing Phrasing due to the presence of phrases such as "in university be" in the error's $Aw{+}_i$ list, there turned out to be few such common phrases in the data set,

[8] This averaging explains why the formula for F does not exactly hold in the Table 6 results.

[9] Per-class results are not shown due to space limitations.

[10] Since parameters for optimizing micro F-score and macro F-score are selected independently, the per-class F-scores associated with micro F-score are different than those used for calculating macro F-score. Hence, when we discuss per-class changes influencing micro F-score, we refer to the former set, and otherwise we refer to the latter set.

[11] All significance tests are paired t-tests, with $p < 0.05$.

so performance on this class remains mostly unchanged. This feature type does, however, result in major improvements to micro and macro performance on Missing Details and Writer Position, the other two classes this feature was designed to help. Indeed, the micro F-score versions of Missing Details and Writer Position improve by 15.3 and 10.8 percentage points respectively. Since these are minority classes, however, the large improvements result in only a small, insignificant improvement in the overall system's micro F-score. The macro F-score results for these classes, however, improve by 6.5% and 17.6% respectively, giving us a nearly 5-point, statistically significant bump in macro F-score after we add this feature.

Confusing Phrasing has up to now stubbornly resisted any improvement, even when we added features explicitly designed to help our system do better on this error type. When we add aggregated **p**art of speech n-gram features on top of the previous system, that changes dramatically. Adding these features makes both our system's F-scores on Confusing Phrasing shoot up almost 8%, resulting in a significant, nearly 4.9% improvement in overall micro F-score and a more modest but insignificant 3.2% improvement in macro F-score (row 5). The micro F-score improvement can also be partly attributed to a four point improvement in Incomplete Prompt Response's micro F-score. The 13.7% macro F-score improvement of the Missing Details error plays a larger role in the overall system's macro F-score improvement than Confusing Phrasing's improvement, however.

The improvement we see in micro F-score when we add aggregated **f**rame features (row 6) can be attributed almost solely to improvements in classification of the minority classes. This is surprising because, as we mentioned before, minority classes tend to have a much smaller impact on overall micro F-score. Furthermore, the overall micro F-score improvement occurrs despite declines in the performances on two of the majority class errors. Missing Details and Writer Position's micro F-score performances increase by 19.1% and 13.4%. The latter is surprising only because of the magnitude of its improvement, as this feature type was explicitly intended to improve its performance. We did not expect this aggregated feature type to be especially useful for Missing Details error identification because very few of these types of features occur in its $Af{+}_i$ list, and there are none in its $Af{-}_i$ list. The few that are in the former list, however, occur fairly often and look like fairly good indicators of this error (both the examples “Event-Event-it” and “Categorization-Item-that” occur in the positive list, and both do seem vague, indicating more details are to be desired).

Overall, this system improves our baseline's macro F-score performance significantly by 13.3% and its micro F-score performance significantly by 10.3%. As we progressed, adding each new feature type to the baseline system, there was no definite and consistent pattern to how the precisions and recalls changed in order to produce the universal increases in the F-scores that we observed for each new system. Both just tended to jerkily progress upward as new feature types were added. This confirms our intuition about these features – namely that they do not all uniformly improve our performance in the same way. Some aim to improve precision by telling us when essays are less likely to be positive instances of an error class, such as any of the $Aw{-}_i$, $Ap{-}_i$, or $Af{-}_i$ features, and others aim to tell us when an essay is more likely to be a positive instance of an error.

6.2 Scoring

Scoring metrics. We design three evaluation metrics to measure the error of our thesis clarity scoring system. The $S1$ metric measures the frequency at which a system predicts the wrong score out of the seven possible scores. Hence, a system that predicts the right score only 25% of the time would receive an $S1$ score of 0.75.

The $S2$ metric measures the average distance between the system's score and the actual score. This metric reflects the idea that a system that estimates scores close to the annotator-assigned scores should be preferred over a system whose estimations are further off, even if both systems estimate the correct score at the same frequency.

Finally, the $S3$ metric measures the average square of the distance between a system's thesis clarity score estimations and the annotator-assigned scores. The intuition behind this metric is that not only should we prefer a system whose estimations are close to the annotator scores, but we should also prefer one whose estimations are not too frequently very far away from the annotator scores. These three scores are given by:

$$\frac{1}{N}\sum_{A_j \neq E'_j} 1, \quad \frac{1}{N}\sum_{i=1}^{N} |A_j - E_j|, \quad \frac{1}{N}\sum_{i=1}^{N} (A_j - E_j)^2$$

where A_j, E_j, and E'_j are the annotator assigned, system estimated, and rounded system estimated scores[12] respectively for essay j, and N is the number of essays.

Results and discussion. Results on scoring are shown in the last three columns of Table 6. We see that the thesis clarity score predicting variation of the **B**aseline system, which employs as features only word n-grams and random indexing features, predicts the wrong score 65.8% of the time. Its predicted score is on average 0.517 points off of the actual score, and the average squared distance between the predicted and actual scores is 0.403.

We observed earlier that a high number of misspellings may be positively correlated with one or more unrelated errors. Adding the **m**isspelling feature to the scoring systems, however, only yields minor, insignificant improvements to their performances under the three scoring metrics.

While adding **k**eyword features on top of this system does not improve the frequency with which the right score is predicted, it both tends to move the predictions closer to the actual thesis clarity score value (as evidenced by the significant improvement in $S2$) and ensures that predicted scores will not too often stray too far from the actual value (as evidenced by the significant improvement in $S3$). Overall, the scoring model employing the **Bmk** feature set performs significantly better than the **B**aseline scoring model with respect to two out of three scoring metrics.

The only remaining feature type whose addition yields a significant performance improvement is the aggregated **w**ord feature type, which improves system **Bmk**'s $S2$ score significantly while having an insignificant impact on the other S metrics.

Neither of the remaining aggregative features yields any significant improvements in performance. This is a surprising finding since, up until we introduced aggregated **p**art-of-speech tag n-gram features into our regressor, each additional feature that helped with error classification made at least a small but positive contribution to at least two out of the three S scores. These aggregative features, which proved to be very powerful when assigning error labels, are not as useful for thesis clarity scoring.

	$S1$ **(Bmkw)**			$S2$ **(Bmkwp)**			$S3$ **(Bmk)**		
Gold	.25	.50	.75	.25	.50	.75	.25	.50	.75
1.0	3.5	3.5	3.5	3.0	3.2	3.5	3.1	3.2	3.3
1.5	2.5	3.0	3.0	2.8	3.1	3.2	2.6	3.0	3.2
2.0	3.0	3.0	3.5	3.0	3.2	3.5	3.0	3.1	3.4
2.5	3.0	3.5	3.5	3.0	3.3	3.6	3.0	3.3	3.5
3.0	3.0	3.5	3.5	3.1	3.4	3.5	3.1	3.3	3.5
3.5	3.5	3.5	4.0	3.2	3.4	3.6	3.2	3.4	3.5
4.0	3.5	3.5	4.0	3.4	3.6	3.8	3.4	3.5	3.7

Table 7: Regressor scores for top three systems.

To more closely examine the behavior of the best scoring systems, in Table 7 we chart the distributions of scores they predict for each gold standard score. As an example of how to read this table, consider the number 2.8 appearing in row 1.5 in the .25 column of the $S2$ (**Bmkwp**) region. This means that 25% of the time, when system **Bmkwp** (which obtains the best $S2$ score) is presented with a test essay having a gold standard score of 1.5, it predicts that the essay has a score less than or equal to 2.8 for the $S2$ metric.

From this table, we see that each of the best systems has a strong bias toward predicting more frequent scores as there are no numbers less than 3.0 in the 50% columns, and about 82.8% of all essays have gold standard scores of 3.0 or above. Nevertheless, no system relies entirely on bias, as evidenced by the fact that each column in the table has a tendency for its scores to ascend as the gold standard score increases, implying that the systems have some success at predicting lower scores for essays with lower gold standard scores.

Finally, we note that the difference in error weighting between the $S2$ and $S3$ scoring metrics appears to be having its desired effect, as there is a strong tendency for each entry in the $S3$ subtable to be less than or equal to its corresponding entry in the $S2$ subtable due to the greater penalty the $S3$ metric imposes for predictions that are very far away from the gold standard scores.

7 Conclusion

We examined the problem of modeling thesis clarity errors and scoring in student essays. In addition to developing these models, we proposed novel features for use in our thesis clarity error model and employed these features, each of which was explicitly designed for one or more of the error types, to train our scoring model. We make our thesis clarity annotations publicly available in order to stimulate further research on this task.

[12] Since our regressor assigns each essay a real value rather than an actual valid thesis clarity score, it would be difficult to obtain a reasonable $S1$ score without rounding the system estimated score to one of the possible values. For that reason, we round the estimated score to the nearest of the seven scores the human annotators were permitted to assign (1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0) only when calculating $S1$.

Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of NSF.

References

Yigal Attali and Jill Burstein. 2006. Automated essay scoring with E-rater v.2.0. *Journal of Technology, Learning, and Assessment*, 4(3).

Jean Carletta. 1996. Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, 22(2):249–254.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. *International Corpus of Learner English (Version 2)*. Presses universitaires de Louvain.

Derrick Higgins and Jill Burstein. 2007. Sentence similarity measures for essay coherence. In *Proceedings of the 7th International Workshop on Computational Semantics*.

Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. 2004. Evaluating multiple aspects of coherence in student essays. In *Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 185–192.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Chapter 11, pages 169–184. MIT Press, Cambridge, MA.

David Jurgens and Keith Stevens. 2010. The S-Space package: An open source package for word space models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 30–35.

Pentti Kanerva, Jan Kristoferson, and Anders Holst. 2000. Random indexing of text samples for Latent Semantic Analysis. In *Proceedings the 22nd Annual Conference of the Cognitive Science Society*, pages 103–106.

Scott Kirkpatrick, C. D. Gelatt, and Mario P. Vecchi. 1983. Optimization by simulated annealing. *Science*, 220(4598):671–680.

Thomas K Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240.

Thomas K. Landauer, Darrell Laham, and Peter W. Foltz. 2003. Automated scoring and annotation of essays with the Intelligent Essay Assessor$^{\text{TM}}$. In *Automated Essay Scoring: A Cross-Disciplinary Perspective*, pages 87–112. Lawrence Erlbaum Associates, Inc., Mahwah, NJ.

Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2009. *English Gigaword Fourth Edition*. Linguistic Data Consortium, Philadelphia.

Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 229–239.

Magnus Sahlgren. 2005. An introduction to random indexing. In *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*.

Mark D. Shermis and Jill C. Burstein. 2003. *Automated Essay Scoring: A Cross-Disciplinary Perspective*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ.

Mark D. Shermis, Jill Burstein, Derrick Higgins, and Klaus Zechner. 2010. Automated essay scoring: Writing assessment and instruction. In *International Encyclopedia of Education (3rd edition)*. Elsevier, Oxford, UK.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420.