Modeling Organization in Student Essays

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

Abstract

Automated essay scoring is one of the most important educational applications of natural language processing. Recently, researchers have begun exploring methods of scoring 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 organization. We present a new annotated corpus and propose heuristic-based and learning-based approaches to scoring essays along the organization dimension, utilizing techniques that involve sequence alignment, alignment kernels, and string kernels.

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). Recent years have seen a surge of interest in this and other educational applications in the NLP community, as evidenced by the panel discussion on “Emerging Application Areas in Computational Linguistics” at NAACL 2009, as well as increased participation in the series of workshops on “Innovative Use of NLP for Building Educational Applications”. Besides its potential commercial value, automated essay scoring brings about a number of relatively less-studied but arguably rather challenging discourse-level problems that involve the computational modeling of different facets of text structure, such as content, coherence, and organization.

A major weakness of many existing essay scoring engines such as IntelliMetric (Elliot, 2001) and Intelligent Essay Assessor (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., 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, and relevance to prompt (Higgins et al., 2004). Automated systems that provide instructional feedback along multiple dimensions of essay quality such as *Criterion* (Burstein et al., 2004) have also begun to emerge.

Nevertheless, there is an essay scoring dimension for which few computational models have been developed — *organization*. Organization refers to the structure of an essay. A high score on organization means that writers introduce a topic, state their position on that topic, support their position, and conclude, often by restating their position (Silva, 1993). A well-organized essay is structured in a way that logically develops an argument. Note that organization is a different facet of text structure than coherence, which is concerned with the transition of ideas at both the global (e.g., paragraph) and local (e.g., sentence) levels. While organization is an important dimension of essay quality, state-of-the-art essay scoring software such as e-rater V.2 (Attali and Burstein, 2006) employs rather simple heuristic-based methods for computing the score of an essay along this particular dimension.

Our goal in this paper is to develop a computational model for the organization of student es-

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 229–239,
MIT, Massachusetts, USA, 9-11 October 2010.

says. While many models of text coherence have been developed in recent years (e.g., Barzilay and Lee (2004), Barzilay and Lapata (2005), Soricut and Marcu (2006), Elsner et al. (2007)), the same is not true for text organization. One reason is the availability of training and test data for coherence modeling. Coherence models are typically evaluated on the sentence ordering task, and hence training and test data can be generated simply by scrambling the order of the sentences in a text. On the other hand, it is not particularly easy to find poorly organized texts for training and evaluating organization models. We believe that student essays are an ideal source of well- and poorly-organized texts. We evaluate our organization model on a data set of 1003 essays annotated with organization scores.

In sum, our contributions in this paper are two-fold. First, we address a less-studied discourse-level task — predicting the organization score of an essay — by developing a computational model of organization, thus establishing a baseline against which future work on this task can be compared. Second, we annotate a subset of our student essay corpus with organization scores and make this data set publicly available. Since progress in organization 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. The essays we used vary greatly in length, containing an average of 31.1 sentences in 7.5 paragraphs, averaging 4.1 sentences per paragraph. About one quarter of the essays had five or fewer paragraphs, and another quarter contained nine or more paragraphs. Similarly, about one quarter of essays contained 24 or fewer sentences and the longest quarter contained 36 or more sentences

We selected a subset consisting of 1003 essays from the ICLE to annotate and use for training and testing of our model of essay organization. While narrative writing asks students to compose descriptive stories, *argumentative* (also known as *persuasive*) writing requires students to state their opinion on a topic and to validate that opinion with convincing arguments. For this reason, we selected only argumentative essays rather than narrative pieces, because they contain the discourse structures and kind of organization we are interested in modeling.

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	147
The prison system is outdated. No civilized society should punish its criminals: it should rehabilitate them.	11	103
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	82

Table 1: Some examples of writing topics.

To ensure representation across native languages of the authors, we selected mostly essays written in response to topics which are well-represented in multiple languages. This avoids many issues that may arise when certain vocabulary is used in response to a particular topic for which essays written by authors from only a few languages are available. Table 1 shows three of the twelve topics selected for annotation. Fifteen native languages are represented in the set of essays selected for annotation.

3 Corpus Annotation

To develop our essay organization model, human annotators scored 1003 essays using guidelines in an essay annotation rubric. Annotators evaluated the organization of each essay using a numerical score from 1 to 4 at half-point increments. 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 organization quality (which can be important in practice), but also makes the prediction task more challenging.

The meaning of each integer score was described and discussed in detail. Table 2 shows the description of each score for the organization dimension.

Score	Description of Essay Organization
4	essay is **well structured** and is organized in a way that logically develops an argument
3	essay is **fairly well structured** but could somewhat benefit from reorganization
2	essay is **poorly structured** and would greatly benefit from reorganization
1	essay is **completely unstructured** and requires major reorganization

Table 2: Descriptions of the meaning of each score.

Our annotators were selected from over 30 applicants who were familiarized with the scoring rubric and given sample essays to score. The six who were most consistent with the expected scores were given additional essays to annotate. To ensure consistency in scoring, we randomly selected a large subset of our corpus (846 essays) to have graded by two different annotators. Analysis of these doubly annotated essays reveals that, though annotators only exactly agree on the organization score of an essay 29% of the time, the scores they apply are within 0.5 points in 71% of essays and within 1.0 point in 93% of essays. Additionally, if we treat one annotator's scores as a gold standard and the other annotator's scores as predictions, the predicted scores have a mean error of 0.54 and a mean squared error of 0.50. Table 3 shows the number of essays that received each of the seven scores for organization.

score	1.0	1.5	2.0	2.5	3.0	3.5	4.0
essays	24	14	35	146	416	289	79

Table 3: Distribution of organization scores.

4 Function Labeling

As mentioned before, a high score on organization means that writers introduce a topic, support their position, and conclude. If one or more of these elements are missing or if they appear out of order (e.g., the conclusion appears before the introduction), the resulting essay will typically be considered poorly organized. Hence, knowing the *discourse function label* of each paragraph in an essay would be helpful for predicting its organization score.

Two questions naturally arise. First, how can we obtain the discourse function label of each paragraph? One way is to automatically acquire such labels from a corpus of student essays where each paragraph is annotated with its discourse function label. To our knowledge, however, there is no publicly available corpus that is annotated with such information. As a result, we will resort to labeling a paragraph with its function label heuristically.

Second, which paragraph function labels would be most useful for scoring the organization of an essay? Based on our linguistic intuition, we identify four potentially useful paragraph function labels: Introduction, Body, Rebuttal, and Conclusion. Table 4 gives the descriptions of these labels.

Label	Name	Paragraph Type
I	Introduction	introduces essay topic and states author's position and main ideas
B	Body	provides reasons, evidence, and examples to support main ideas
C	Conclusion	summarizes and concludes arguments made in body paragraphs
R	Rebuttal	considers counter-arguments to thesis or main ideas

Table 4: Descriptions of paragraph function labels.

Setting aside for the moment the problem of exactly how to predict an essay's organization score given its paragraph sequence, the problem of obtaining paragraph labels to use for this task still remains. As mentioned above, we adopt a heuristic approach to paragraph function labeling. The question, then, is: what kind of knowledge sources should our heuristics be based on? We have identified two types of knowledge sources that are potentially useful for paragraph labeling. The first of these are positional, dealing with where in the essay a paragraph appears. So for example, the first paragraph in an essay is likely to be an Introduction, while the last is likely to be a Conclusion. A paragraph in any other position, on the other hand, is more likely to be a Body or Rebuttal paragraph.

Label	Name	Sentence Function
P	Prompt	restates the prompt given to the author and contains no new material or opinions
T	Transition	shifts the focus to new topics but contains no meaningful information
H	Thesis	states the author's position on the topic for which he/she is arguing
M	Main Idea	asserts reasons and foundational arguments that support the thesis
E	Elaboration	further explains reasons and ideas but contains no evidence or examples
S	Support	provides evidence and examples to support the claims made in other statements
C	Conclusion	summarizes and concludes the entire argument or one of the main ideas
R	Rebuttal	considers counter-arguments that contrast with the thesis or main ideas
O	Solution	puts to rest the questions and problems brought up by counter-arguments
U	Suggestion	proposes solutions the problems brought up by the argument

Table 5: Descriptions of sentence function labels.

A second potentially useful knowledge source involves the types of sentences appearing in a paragraph. This idea presupposes that, like paragraphs, sentences too can have discourse function labels indicating the logical role they play in an argument. The sentence label schema we propose, which is described in Table 5, is based on work in discourse structure by Burstein et al. (2003), but features additional sentence labels.

To illustrate why these sentence function labels may be useful for paragraph labeling, consider a paragraph containing a Thesis sentence. The presence of a Thesis sentence is a strong indicator that the paragraph containing it is either an Introduction or Conclusion. Similarly, a paragraph containing Rebuttal or Solution sentences is more likely to be a Body or Rebuttal paragraph.

Hence, to obtain a paragraph's function label, we need to first label its sentences. However, we are faced with the same problem: how can we obtain the sentence function labels? One way is to learn them from a corpus where each sentence is manually annotated with its sentence function label, which is the approach adopted by Burstein et al. (2003). However, this annotated corpus is not publicly available. In fact, to our knowledge, there is no publicly-available corpus that is annotated with sentence function labels. Consequently, we adopt a heuristic approach to sentence function labeling.

Overall, we created a knowledge-lean set of heuristic rules labeling paragraphs and sentences. Because many of the paragraph labeling heuristics depend on the availability of sentence labels, we will describe the sentence labeling heuristics first. For each sentence function label x, we identify several features whose presence increases our confidence that a given sentence is an example of x. So for example, the presence of any of the words "agree", "think", or "opinion" increases our confidence that the sentence they occur in is a Thesis. If the sentence instead contains words such as "however", "but", or "argue", these increase our confidence that the sentence is a Rebuttal. The features we examine for sentence labeling are not limited to words, however. Each content word the sentence shares with the essay prompt gives us evidence that the sentence is a restatement of the prompt. Having searched a sentence for all these clues, we finally assign the sentence the function label having the most support among the clues found.

The heuristic rules for paragraph labeling are similar in nature, though they depend heavily on the labels of a paragraph's component sentences. If a paragraph contains Thesis, Prompt, or Background sentences, the paragraph is likely to be an Introduction. However, if a paragraph contains Main Idea, Support, or Conclusion sentences, it is likely to be a Body paragraph. Finally, as mentioned previously, some positional information is used in labeling paragraphs. For example, a paragraph that is the first paragraph in an essay is likely to be an Introduction, but a paragraph that is neither the first nor the last is likely to be either a Rebuttal or Body paragraph. After searching a paragraph for all these features, we gather the pieces of evidence in support of each paragraph label and assign the paragraph the label having the most support.[1]

[1] Space limitations preclude a complete listing of these para-

5 Heuristic-Based Organization Scoring

Having applied labels to each paragraph in an essay, how can we use these labels to predict the essay's score? Recall that the importance of each paragraph label stems not from the label itself, but from the sequence of labels it appears in. Motivated by this observation, we exploit a technique that is commonly used in bioinformatics — *sequence alignment*. While sequence alignment has also been used in text and paraphrase generation (e.g., Barzilay and Lee (2002; 2003)), it has not been extensively applied to other areas of language processing, including essay scoring. In this section, we will present two heuristic approaches to organization scoring, one based on aligning *paragraph sequences* and the other on aligning *sentence sequences*.

5.1 Aligning Paragraph Sequences

As mentioned above, our first approach to heuristic organization scoring involves aligning paragraph sequences. Specifically, this approach operates in two steps. Given an essay e in the test set, we (1) find the k essays in the training set that are most similar to e via paragraph sequence alignment, and then (2) predict the organization score of e by aggregating the scores of its k nearest neighbors obtained in the first step. Below we describe these two steps in detail.

First, to obtain the k nearest neighbors of e, we employ the Needleman-Wunsch alignment algorithm (Needleman and Wunsch, 1970), which computes a similarity score for any pair of essays by finding an optimal alignment between their paragraph sequences. To illustrate why we believe sequence alignment can help us determine which essays are most similar, consider two example essays. One essay, which we will call IBBBC, begins with an Introductory paragraph, follows it with three Body paragraphs, and finally ends with a Concluding paragraph. Another essay CRRRI begins with a paragraph stating its Conclusion, follows it with three Rebuttal paragraphs, and ends with a paragraph Introducing the essay's topic. We can tell by a casual glance at the sequences that any reasonable similarity function should tell us that they are not very similar. The Needleman-Wunsch alignment algorithm has this effect since the score of the alignment it produces would be hurt by the facts that (1) there is not much overlap in the sets of paragraph labels each contains, and (2) the paragraph labels they do share (I and C) do not occur in the same order. The resulting alignment would therefore contain many mismatches or indels.[2]

If we now consider a third essay whose paragraph sequence could be represented as IBRBC, a good similarity function should tell us that IBBBC and IBRBC are very similar. The Needleman-Wunsch alignment score between the two paragraph sequences has this property, as the alignment algorithm would discover that the two sequences are identical except for the third paragraph label, which could be mismatched for a small penalty. We would therefore conclude that the IBBBC and IBRBC essays should receive similar organization scores.

To fully specify how to find the k nearest neighbors of an essay, we need to define a similarity function between paragraph labels. In sequence alignment, similarity function $S(i, j)$ tells us how likely it is that symbol i (in our case, a paragraph label) will be substituted with another symbol j. While we expect that in an alignment between high-scoring essays, an Introduction paragraph is most likely to be aligned with another Introduction paragraph, how much worse should the alignment score be if an Introduction paragraph needs to be mismatched with a Rebuttal paragraph or replaced with an indel? We solve this problem by heuristically defining the similarity function as follows: $S(i, j) = 1$ when $i = j$, $S(i, j) = -1$ when $i \neq j$, and also $S(i, -) = S(-, i) = -1$, where '$-$' is an indel. In other words, the similarity function encourages the alignment between two identical function labels and discourages the alignment between two different function labels, regardless of the type of function labels.

After obtaining the k nearest neighbors of e, the next step is to predict the organization score of e by aggregating the scores of its k nearest neighbors into one number. (Note that we know the organiza-

graph and sentence labeling heuristics. See our website at http://www.hlt.utdallas.edu/~alan/ICLE/ for the complete list of heuristics.

[2] In pairwise sequence alignment, a mismatch occurs when one symbol has to be substituted for another to make two sequences match. An indel indicates that in order to transform one sequence to match another, we must either **in**sert a symbol into one sequence or **del**ete a symbol from the other sequence.

tion score of each nearest neighbor, since they are all taken from the training set.) One natural way to do this would be to take the mean, median, or mode of its k nearest neighboring essays from the training set. Hence, our first heuristic method H_p for scoring organization has three variants.

5.2 Aligning Sentence Sequences

An essay's paragraph sequence captures information about its organization at a high level, but ignores much of its lower level structure. Since we have also heuristically labeled sentences, it now makes sense to examine the sequences of sentence function labels within an essay's paragraphs. The intuition is that at least some portion of an essay's organization score can be attributed to the organization of the sentence sequences of its component paragraphs.

To address this concern, we propose a second heuristic approach to organization scoring. Given a test essay e, we first find for each *paragraph* in e the k *paragraphs* in the training set that are most similar to it. Specifically, each paragraph is represented by its sequence of *sentence* function labels. Given this paragraph representation, we can find the k nearest neighbors of a paragraph by applying the Needleman-Wunsch algorithm described in the previous subsection to align *sentence* sequences, using the same similarity function we defined above.

Next, we score each paragraph p_i by aggregating the scores of its k nearest neighbors obtained in the first step, assuming the score of a nearest neighbor paragraph is the same as the organization score of the training set essay containing it. As before, we can employ the mean, median, or mode to aggregate the scores of the nearest neighbors of p_i.

Finally, we predict the organization score of e by aggregating the scores of its paragraphs obtained in the second step. Again, we can employ mean, median, or mode to aggregate the scores. Since we have three ways of aggregating the scores of a paragraph's nearest neighbors and three ways of aggregating the resulting paragraph scores, this second method H_s for scoring organization has nine variants.

6 Learning-Based Organization Scoring

In the previous section, we proposed two heuristic approaches to organization scoring, one based on aligning paragraph label sequences and the other based on aligning sentence label sequences. In the process of constructing these two systems, however, we created a lot of information about the essays which might also be useful for organization scoring, but which the heuristic systems are unable to exploit. To remedy the problem, we introduce three learning-based systems which abstract the additional information we produced in three different ways. In each system, we use the SVM^{light} (Joachims, 1999) implementation of regression support vector machines (SVMs) (Cortes and Vapnik, 1995) to train a regressor because SVMs have been frequently and successfully applied to a variety of NLP problems.

6.1 Linear Kernel

Owing to the different ways we presented of combining the scores of an essay's nearest neighbors, the paragraph label sequence alignment approach has three variants, and its sentence label sequence alignment counterpart has nine. Unfortunately, these heuristic approaches suffer from two major weaknesses. First, it is not intuitively clear which of these 12 ways for predicting an essay's organization score is clearly better than the others. Second, it is not clear that the k nearest neighbors of an essay will always be similar to it with respect to organization score. While we do expect the alignment scores between good essays with reasonable paragraph sequences to be high, poorly organized essays by their nature have more random paragraph sequences. Hence, we have no intuition about the k nearest neighbors of a poor essay, as it may have as high an alignment score with another poorly organized essay as with a good essay.

Our solution to these problems is to use the organization scores obtained by the 12 heuristic variants as features in a linear kernel SVM learner. We believe that using the estimates given by all the 12 variants of the two heuristic approaches rather than only one of them addresses the first weakness mentioned above. The second weakness, on the other hand, is addressed by treating the organization score predictions obtained by the nearest neighbor methods as features for an SVM learner rather than as estimates of an essay's organization score.

The approach we have just described, however, does not exploit the full power of linear kernel

SVMs. One strength of linear kernels is that they make it easy to incorporate a wide variety of different types of features. In an attempt to further enhance the prediction capability of the SVM regressor, we will provide it with not only the 12 features derived from the heuristic-based approaches, but also with two additional types of features.

First, to give our learner more direct access to the information we used to heuristically predict essay scores, we can extract *paragraph label subsequences*[3] from each essay and use them as features. To illustrate the intuition behind these features, consider two paragraph subsequences: Introduction–Body and Rebuttal–Introduction. It is fairly typical to see the first subsequence, I–B, at the beginning of a good essay, so its occurrence should give us a small amount of evidence that the essay it occurs in is well-organized. The presence of the second subsequence, R–I, however, should indicate that its essay's organization is poor because, in general, a good essay should not give a Rebuttal before an Introduction. Because we can envision subsequences of various lengths being useful, we create a binary presence or absence feature in the linear kernel for each paragraph subsequence of length 1, 2, 3, 4, or 5 appearing in the training set.

Second, we employ *sentence label subsequences* as features in the linear kernel. Recall that when describing our alignment-based nearest neighbor organization score prediction methods, we noted that an essay's organization score may be partially attributable to how well the sentences within its paragraphs are organized. For example, if one of an essay's paragraphs contains the sentence label subsequence Main Idea–Elaboration–Support–Conclusion this gives us some evidence that the essay is overall well-organized since one of its component paragraphs contains this reasonably-organized subsequence. An essay with a paragraph containing the subsequence Conclusion–Support–Thesis–Rebuttal, however, is likely to be poorly organized because this is a poorly-organized subsequence. Since sentence label subsequences of differing lengths may be useful for score prediction, we create a binary presence or absence feature for each sentence label subsequence of length 1, 2, 3, 4, or 5 in the training set.

While the number of nearest neighbor features is manageable, the presence of a large number of features can sometimes confuse a learner. For that reason, we do feature selection on the two types of subsequence features, selecting only 100 features for each type that has the highest information gain (see Yang and Pedersen (1997) for details). We call the system resulting from the use of these three types of features Rl_{nps} because it uses **R**egression with **l**inear kernel to predict essay scores, and it uses **n**earest neighbor, **p**aragraph subsequence, and **s**entence subsequence features.

6.2 String Kernel

In a traditional learning setting, the feature set employed by an off-the-shelf learning algorithm typically consists of *flat* features (i.e., features whose values are discrete- or real-valued, as the ones described in the Linear Kernel subsection). Advanced machine learning algorithms such as SVMs, on the other hand, have enabled the use of *structured* features (i.e., features whose values are structures such as parse trees and sequences), owing to their ability to employ *kernels* to efficiently compute the similarity between two potentially complex structures.

Perhaps the most obvious advantage of employing structured features is *simplicity*. To understand this advantage, consider learning in a traditional setting. Recall that we can only employ flat features in this setting, as we did with the linear kernel. Hence, if we want to use information from a parse tree as features, we will need to design heuristics to extract the desired parse-based features from parse trees. For certain tasks, designing a good set of heuristics can be time-consuming and sometimes difficult. On the other hand, SVMs enable a parse tree to be employed directly as a structured feature, obviating the need to design heuristics to extract information from potentially complex structures. However, structured features have only been applied to a handful of NLP tasks such as semantic role labeling (Moschitti, 2004), syntactic parsing and named entity identification (Collins and Duffy, 2002), relation extraction (Bunescu and Mooney, 2005), and coreference resolution (Versley et al., 2008). Our goal here is to explore this rarely-exploited capability of SVMs for the task of essay scoring.

[3] Note that a subsequence is not necessarily contiguous.

While the vast majority of previous NLP work on using structured features have involved tree kernels, we employ a kernel that is rarely investigated in NLP: *string kernels* (Lodhi et al., 2002). Informally, a string kernel aims to efficiently compute the similarity between two strings (or sequences) of symbols based on the similarity of their subsequences. We apply string kernels to essay scoring as follows: we represent each essay using its paragraph function label sequence, and employ a string kernel to compute the similarity between two essays based on this representation. Typically, a string kernel takes as input two parameters: K (which specifies the length of the subsequences in the two strings to compare) and λ (which is a value between 0 and 1 that specifies whether matches between non-contiguous subsequences in the two strings should be considered as important as matches between contiguous subsequences). In our experiments, we select values for these parameters in a somewhat arbitrary manner. In particular, since λ ranges between 0 and 1, we simply set it to 0.5. For K, since in the flat features we considered all paragraph label sequences of lengths from 1 to 5, we again take the middle value, setting it to 3. We call the system using this kernel Rs because it uses a **R**egression SVM with a **s**tring kernel to predict essay scores.

6.3 Alignment Kernel

In general, the purpose of a kernel function is to measure the similarity between two examples. The string kernel we described in the previous subsection is just one way of measuring the similarity of two essays given their paragraph sequences. While this may be the most obvious way to use paragraph sequence information from a machine learning perspective, our earlier use of the Needleman-Wunsch algorithm suggests a more direct way of extracting structured information from paragraph sequences.

More specifically, recall that the Needleman-Wunsch algorithm finds an optimal alignment between two paragraph sequences, where an optimal alignment is defined as an alignment having the highest possible alignment score. The optimal alignment score can be viewed as another similarity measure between two essays. As such, with some slight modifications, the alignment score between two paragraph sequences can be used as the kernel value for an Alignment Kernel.[4] We call the system using this kernel Ra because it uses a **R**egression SVM with an **a**lignment kernel to predict essay scores.

6.4 Combining Kernels

Recall that the flat features are computed using a linear kernel, while the two types of structured features are computed using string and alignment kernels. If we want our learner to make use of more than one of these types of features, we need to employ a *composite* kernel to combine them. Specifically, we define and employ the following composite kernel:

$$K_c(F_1, F_2) = \frac{1}{n} \sum_{i=1}^{n} K_i(F_1, F_2),$$

where F_1 and F_2 are the full set of features (containing both flat and structured features) that represent the two essays under consideration, K_i is the ith kernel we are combining, and n is the number of kernels we are combining. To ensure that each kernel under consideration contributes equally to the composite kernel, each kernel value $K_i(F_1, F_2)$ is normalized so that its value falls between 0 and 1.

7 Evaluation

7.1 Evaluation Metrics

We designed three evaluation metrics to measure the error of our organization scoring system. The simplest metric, $S1$, is perhaps the most intuitive. It 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 is slightly less intuitive than $S1$, but no less reasonable. It 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 $S3$ evaluation metric measures the average square of the distance between a system's

[4] In particular, we note that for theoretical reasons, a kernel function must always return a non-negative value. The alignment score function does not have this property, so we increase all alignment scores until their theoretical minimum value is 0.

organization score estimations and the annotator-assigned scores. The intuition behind this system 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:

$$\frac{1}{N}\sum_{A_i \neq E_i} 1, \quad \frac{1}{N}\sum_{i=1}^{N} |A_i - E_i|, \quad \frac{1}{N}\sum_{i=1}^{N} (A_i - E_i)^2,$$

where A_i and E_i are the annotator assigned and system estimated scores respectively for essay i, and N is the number of essays. Since many of the systems we have described assign test essays real-valued organization scores, to obtain E_i for system $S1$ we round the outputs of each system 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).

To test our system, we performed 5-fold cross validation on our 1003 essay set, micro-averaging our results into three scores corresponding to the three scoring metrics described above.

7.2 Results and Discussion

The average baseline. As mentioned before, there is no standard baseline for organization modeling against which we can compare our systems. To start with, we employ a simple "average" baseline. Avg computes the average organization score of essays in the training set and assigns this score to each test set essay. Results of this baseline are shown in row 1 of Table 6. Though simple, this baseline is by no means easy-to-beat, since 41% of the essays have a score of 3, and 96% of the essays have a score that is within one point of 3.

Heuristic baselines. Recall that we have 12 versions of the two heuristic approaches to organization prediction. Space limitations preclude a discussion of the results of all these versions, so instead, to obtain the strongest baseline results, we show only the best results achieved by the three versions based on aligning paragraph label sequences in row 2 (H_p) and the best results achieved by the nine versions based on aligning sentence label sequences in row 3 (H_s) of Table 6. It is clear from the results that the H_p systems yielded the best baseline predictions under all three scoring metrics, performing significantly better than both the Avg and H_s systems ($p < 0.01$) with respect to the $S2$ and $S3$ metrics, but its $S1$ performance is less significant with respect to Avg ($p < 0.1$) and is indistinguishable at even the $p < 0.1$ level from H_s.[5] In general, however, it appears to be the case that systems based on aligning paragraph label sequences achieve better results than systems that attempt to align sentence label sequences.

	System	$S1$	$S2$	$S3$
1	Avg	.585	.412	.348
2	H_p	.548	.339	.198
3	H_s	.575	.397	.329
4	Rl_{nps}	.520	.331	.186
5	Rs	.577	.369	.222
6	Ra	.686	.519	.429
7	Rls_{nps}	.534	.332	.187
8	Rla_{nps}	.541	.332	.178
9	Rsa	.517	.325	.177
10	$Rlsa_{nps}$	.517	.323	.175

Table 6: System Performance

Learning-based approaches. Rows 4–6 of Table 6 show the results we obtained using each of the three single-kernel systems. When compared to the best baseline, these results suggest that H_p is a pretty good heuristic approach to organization scoring. In fact, only one of these three learning-based systems (Rl_{nps}) performs better than H_p under the three scoring metrics, and in each case, the difference between the two is not significant even at $p < 0.1$. This suggests that, even though Rl_{nps} performs slightly better than H_p, the only major benefit we have obtained by using the linear kernel is that it has made it unnecessary for us to choose between the 12 proposed heuristic systems.

Considering that the second best one-kernel system, Rs, does not have access to any of the nearest neighbor features, which have already proven useful, its performance seems reasonably good in that its performance is at least better than the Avg system. This suggests that, even though Rs does not perform exceptionally, it is extracting some useful information for organization scoring from the heuristically assigned paragraph label sequences. The best one-kernel system, Rl_{nps}, however, is sig-

[5] All significance tests are two-tailed paired t-tests.

nificantly better than Rs with respect to all three scoring metrics, with $p < 0.1$ for $S1$ and $p < 0.05$ for $S2$ and $S3$. By contrast, it initially appears that the alignment kernel is not extracting any useful information from these paragraph sequences at all, since its $S1$, $S2$, and $S3$ scores are all much worse than all of the baseline systems. The second best one-kernel system Rs performs significantly better than Ra at $p < 0.01$ for all three scoring metrics.

Next, we explore the impact of composite kernels, which allow our learners to make use of multiple types of flat and structured features. Specifically, the results shown in rows 7–9 are obtained by combining two kernels at a time. These experiments reveal the surprising result that the two worst performing single-kernel systems, Rs and Ra, when combined into Rsa, yield the best two-kernel system results, which are significant with respect to the best one-kernel system results under $S3$ at $p < 0.1$. This result suggests that these two different methods of extracting information from paragraph sequences provide us with different kinds of evidence useful for organization scoring, although neither method by itself was exceptionally useful. Though Rsa does not have any access to nearest neighbor information, it still performs significantly better than H_p at $p < 0.05$ under $S1$ and $S3$.

While we have already pointed out that Rsa is the best composite two-kernel system, it is not clear which of Rls_{nps} and Rla_{nps} is second-best. Neither system consistently performs better than the other under all three scoring metrics, and the differences between them are not significant even at $p < 0.1$. It is clear only that Rsa is better than both, as its scores are statistically significantly better at $p < 0.01$ with respect to Rls_{nps} and Rla_{nps} under at least one of the three scoring metrics in each case.

Finally, in the last row of Table 6, we combine all three kernels into one SVM learner. The most important lesson we learn from this experiment is that each of the three kernels provides the learner with a different kind of useful information, so that a composite kernel using all three sources of information performs better than any system using fewer kernels. Although the improvements over the best two-kernel system (Rsa) and one-kernel system (Rl_{nps}) are small, they are still statistically significant at $p < 0.1$ under one of the scoring metrics, $S3$. When we compare this combined system to the best baseline (H_p), we discover the improvements derived from the three-kernel system are significant improvements over it at $p < 0.05$ and $p < 0.01$ with respect to $S1$ and $S3$ respectively.

Feature analysis. To better understand which of the three flat features (nearest neighbors, paragraph label sequences, or sentence label sequences) contributes the most to the linear kernel portion of the systems' performances, we analyze the three feature types on Rl_{nps} using the backward elimination feature selection algorithm. First, we remove each of the three feature groups independently from the Rl_{nps}'s feature set and determine which of the three removals yields the best performance according to each scoring metric. Next, among the remaining two feature groups, we repeat the same step, removing each of the two groups independently from the feature set to determine which of the two removals yields the best performance.

While space limitations preclude showing the actual numbers, the trend is consistent among all three scoring metrics: the first feature type to remove is paragraph sequences (meaning that they are the least important) and the last to remove is the nearest neighbor features. Nevertheless, performance always drops when a feature type is removed, indicating that all three feature types contribute positively to overall performance. The fact that flat paragraph sequence features proved to be least useful highlights the importance of the structured methods we presented for using paragraph sequence information.

8 Conclusions

We have investigated the relatively less-studied problem of modeling the organization in student essays. The contributions of our work include the novel application of two techniques from bioinformatics and machine learning — sequence alignment and string kernels, as well as the introduction of alignment kernels — to essay scoring. We showed that each technique makes a significant contribution to a scoring system, and we hope that this work will increase awareness of these powerful techniques among NLP researchers. Finally, to stimulate work on this problem, we make our corpus of annotated essays available to other researchers.

Acknowledgments

We thank the three reviewers for their comments. Our six annotators, Andrew Hubbs, Karin Khoo, Jayne Koath, Christopher Maier, Andrew Mallon, and Cory Thornton, all deserve numerous thanks, because without the countless hours they each spent annotating hundreds of essays, none of the research described in this paper would have been possible.

References

Yigal Attali and Jill Burstein. 2006. Automated essay scoring with e-rater V.2. *Journal of Technology, Learning, and Assessment*, 4(3).

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the ACL*, pages 141–148.

Regina Barzilay and Lillian Lee. 2002. Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of EMNLP*, pages 164–171.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL 2003: Main Proceedings*, pages 16–23.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Main Proceedings*, pages 113–120.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT/EMNLP*, pages 724–731.

Jill Burstein, Martin Chodorow, and Claudia Leacock. 2004. Automated essay evaluation: The *Criterion* online writing evaluation service. *AI Magazine*, 25(3), 27–36.

Jill Burstein, Daniel Marcu, and Kevin Knight. 2003. Finding the write stuff: Automatic identification of discourse structure in student essays. *IEEE Intelligent Systems*, 18(1):32–39.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the ACL*, pages 263–270.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Scott Elliot. 2001. IntelliMetric: From here to validity. Paper presented at the annual meeting of the American Educational Research Association, Seattle, WA.

Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *NAACL HLT 2007: Proceedings of the Main Conference*, pages 436–443.

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

Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. 2004. Evaluating multiple aspects of coherence in student essays. In *HLT-NAACL 2004: Main Proceedings*, pages 185–192.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In Bernhard Scholkopf and Alexander Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 44–56. MIT Press.

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

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Christopher J. C. H. Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

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

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the ACL*, pages 335–342.

Saul Ben Needleman and Christian Dennis Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, March.

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

Mark Shermis, Jill Burstein, Derrick Higgins, and Klaus Zechner. 2010. Automated essay scoring: Writing assessment and instruction. In *International Encyclopedia of Education (3rd edition)*, pages 20–26.

Tony Silva. 1993. Toward an understanding of the distinct nature of L2 writing: The ESL research and its implications. 27(4):657–677.

Radu Soricut and Daniel Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 803–810.

Yannick Versley, Alessandro Moschitti, Massimo Poesio, and Xiaofeng Yang. 2008. Coreference systems based on kernels methods. In *Proceedings of COLING*, pages 961–968.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of ICML*, pages 412–420.