SENTENCE SIMILARITY ANALYSIS WITH APPLICATIONS IN AUTOMATIC SHORT ANSWER GRADING

Michael A.G. Mohler

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

August 2012

Mohler, Michael A.G. *Sentence similarity analysis with applications in automatic short answer grading*. Doctor of Philosophy (Computer Science and Engineering), August 2012, 80 pp., 19 tables, 5 figures, bibliography, 107 titles.

In this dissertation, I explore unsupervised techniques for the task of automatic short answer grading. I compare a number of knowledge-based and corpus-based measures of text similarity, evaluate the effect of domain and size on the corpus-based measures, and also introduce a novel technique to improve the performance of the system by integrating automatic feedback from the student answers. I continue to combine graph alignment features with lexical semantic similarity measures and employ machine learning techniques to show that grade assignment error can be reduced compared to a system that considers only lexical semantic measures of similarity. I also detail a preliminary attempt to align the dependency graphs of student and instructor answers in order to utilize a structural component that is necessary to simulate human-level grading of student answers. I further explore the utility of these techniques to several related tasks in natural language processing including the detection of text similarity, paraphrase, and textual entailment.

ACKNOWLEDGEMENTS

I dedicate this dissertation to my wife, Megan, who has stood by me throughout. You are my partner, my drive, and my foil. I thank God for bringing you into my life, and I thank you for your patience, you labors, and your love.

This material is based upon work supported by the National Science Foundation under Grant No. 0747340. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

TABLE OF CONTENTS

<center>LIST OF TABLES</center>

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Many problems in natural language processing (NLP) require, as a subtask, the ability to estimate the similarity between two pieces of text. For instance, modern information retrieval is predicated upon the need to rank a large set of documents (e.g. websites) based upon their similarity to a user's query string. Likewise, automatic summarization, text classification, information extraction, and automatic translation evaluation rely upon a system's knowledge of the underlying similarity between two texts. The task I focus on here, the automatic grading of short student answers (or computer-aided assessment), is another example, which is becoming increasingly important as online education becomes more and more common.

In this thesis, I present my work in building a system capable of automatically grading student responses to short-answer questions and attempt to show the applicability of these methods beyond computer-aided assessment (CAA) to the detection of textual similarity more generally. Unlike previous work, which has either required the availability of manually crafted patterns [70, 96], or large sets of training data to bootstrap such patterns [80], I have attempted to devise a system which is not pattern-based but rather uses lexical, syntactic, and semantic similarity techniques to determine a appropriate score.

I have explicitly addressed the short answer grading task as a textual similarity problem. With that in mind, I have employed several existing and well-known bag-of-words similarity measures. In order to address the limitations associated with the bag-of-words paradigm, I have attempted to enrich this model with more complex dependency graph subsumption-based measures of similarity inspired by recent work in textual entailment [40, 65, 87]. I go on to produce a hybrid system which combines the above techniques for assessing similarity using one of several flavors of support vector machine (SVM) learning in order to exploit the best measures available.

Furthermore, I have engaged in a portability study, in which I apply this grading system to the related tasks of semantic text similarity, paraphrase detection, and recognizing textual entailment. I use several of the most commonly cited datasets for each of these tasks and compare my results against existing state-of-the-art systems.

Over the course of this work, I have sought and found answers to the following questions. First, given a number of corpus-based and knowledge-based methods previously proposed for word and text semantic similarity, what are the measures that work best for the task of short answer grading? Second, given a corpus-based measure of similarity, what is the impact of the domain and the size of the training corpus on the utility of the measure? Third, to what extent is it possible to enhance the quality of the grading system by supplementing the gold-standard answer with the answers of other students? Fourth, does the dependency parse structure of a text provide clues that can be exploited to improve upon existing BOW methodologies for short answer grading? Fifth, to what extent can machine learning be applied to improve upon existing approaches to short answer grading? Finally, can the methodologies I have proposed for short answer grading be successfully used to detect textual similarity, paraphrase, and entailment?

1.1. Organization

The remainder of this thesis is organized as follows. Chapter 2 provides the theoretical background in both textual similarity (generally speaking) and grading as it relates to student assessment. I then outline existing work in the fields of short answer grading, text similarity, paraphrase detection, and textual entailment, especially those which have had a substantial impact upon the design of this system. Chapter 3 describes in detail the datasets that I have used in earlier work and provides an introduction to the datasets I have used in the portability study. Chapter 3 also contains a brief discussion on the applicability of various evaluation metrics for the tasks of short answer grading, text similarity, paraphrase detection, and recognizing textual entailment. Chapter 4 details the contributions of my work focusing on the bag-of-words and pseudo-relevance feedback techniques which were introduced [72] at the 2009 European Association for Computational Linguistics (EACL) conference in Athens, Greece. Chapter 5 describes the usage of both dependency graph alignment techniques and machine learning to supplement the simpler bag-of-words techniques [71] and includes an analysis of the sources of error associated with the current system. In Chapter 6, I describe my work as applied to the "Semantic Text Similarity" (STS) task of the 2012 Semantic Evaluation Workshop (SemEval 2012) as well as the application of the system to the tasks of text similarity, paraphrase detection, and recognizing textual entailment using other well-known datasets. Finally, in Chapter 7 I discuss my findings.

CHAPTER 2

BACKGROUND

In this chapter, I show how this work may be placed into the larger context of computer-aided assessment (CAA) and similarity detection research by exploring the practical and theoretical justifications for this type of endeavor and the work of others who have wandered along a similar path. The remainder of this chapter is organized as follows. In Section 2.1 I discuss what it means for two things to be similar by digging into the psychological underpinnings of of similarity as seen by humans. In Section 2.2, I look at the act of grading in its own right by tracing the evolution of scholastic assessment as well as groundbreaking efforts to automate the process. Finally, in Section 2.3 I look at the efforts of other researchers in the fields of CAA, textual similarity, paraphrase, and entailment who have paved the way for this present work.

2.1. Theory of Similarity

Humans are good at determining when two things are similar, but it is a very difficult trait to quantify or to defend. An extended quote from the American philosopher Nelson Goodman's "Seven strictures on similarity" [38] may be illustrative:

> When, in general, are two things similar? The first response is likely to be: "When they have at least one property in common." But since every two things have some property in common, this will make similarity a universal and hence useless relation. That a given two things are similar will hardly be notable news if there are no two things that are not similar.
>
> Are two things similar, then, only if they have all their properties in common? This will not work either; for of course no two things have all their properties in common. Similarity so interpreted will be an empty and hence useless relation. That a given two things are similar in this sense would be notable news indeed, but false.
>
> By now we may be ready to settle for a comparative rather than a categorical formula. Shall we say that two things *a* and *b* are more alike than two others *c* and *d* if *a* and *b* have more properties in common than do *c* and *d*... More to the point would be counting not all shared properties but rather only *important* properties – or better, considering not the count but the overall importance of the shared properties. Then *a* and *b* are more alike than *c* and *d* if the cumulative importance of the properties shared by *a* and *b* is greater than that of the properties shared by *c* and *d*. But importance is a highly volatile matter, varying with every shift of context and interest, and quite

incapable of supporting the fixed distinctions that philosophers so often seek to rest upon it.

Despite decades or more of research among psychologists [7], there remains no agreed upon model for human cognition of similarity between objects. It would appear that transitivity does not hold among similarity relations and that there is some asymmetry where A is perceived to be more similar to B than B is to A [99]. Likewise, similarity is not the same thing as relatedness. Paul Resnik illustrates this by suggesting that a car and gasoline are very more closely related than a car and a bicycle, though the latter two are more similar [83]. What this all adds up to is that any research involving similarity detection in artificial intelligence has to cope with the nebulous psychological models that they are attempting to mimic. This is no less true for similarity in natural language processing (NLP).

A wide variety of tasks in natural language processing require some degree of similarity detection at various levels of granularity. At the most primitive stage, systems for lexical substitution, synonymy detection, and text generation need to be able to determine whether two words are similar enough to be used interchangeably. At the other extreme, document classification and clustering assess the similarity between two documents (or between a document and a document model). Perhaps the most ubiquitous example of textual similarity being used today is between a short query and the document within a collection. This is the foundation of the entire search engine industry without which our collective Internet activity would be limited to a set of bookmarks and hyperlinks. The general task that I am most concerned with in this work involves quantifying the similarity between two short pieces of text – at the level of a phrase, sentence, or paragraph – henceforth called *sentence similarity*. However, the various levels can not be entirely disentangled (sentences are made of words, documents are made of sentences), so it may be instructive to first consider similarity that is not at the level of sentences.

Term (or word) similarity is a well-studied field in its own right with most research measuring the distance between words in a thesaurus or lexical ontology [31, 79]. Obviously, detecting similarity at the phrase or sentence level requires a system to be able to recognize the compositional words and what meanings they are capable of conveying individually. Beyond that, though, there must also be some means of analyzing how the structure of the phrasing affects the overall meaning. For instance, the two phrases "a cat in a hat" and "a hat in a cat" represent a complete lexical overlap, but describe two very different problems.

Quantifying the similarity of full documents is much more of a statistical endeavor. Not every word in one document needs to be found in the other, and not every sentence in one document requires a direct complement. Similarity at this level is better defined by likeness of topic, style, or structure [16, 13]. In most modern work, document-similarity (or query-document similarity) is measured by converting each document into a vector-space representation based upon frequency of word choice and usually some means of weighting the importance of each dimension [90, 89]. The two vectors are then compared using standard vector similarity measures (e.g. Euclidean distance, cosine similarity, etc.).

Note that in neither case (word- or document-similarity) do the measurements depend upon any analysis of individual sentences or phrases. They are neither concerned with the syntax itself, nor the semantic differences that can only be indicated through a change in syntactical structure. Textual analysis of this type is in the purview of sentence similarity and, until recently, comparatively little research has been devoted to it.

2.2. Grading Background

Fundamentally, the entire discipline of education has as its goal that a student increase in knowledge, and there are many tools at the disposal of educators as they attempt to achieve this goal – lectures, assignments, projects, hands-on work, face-to-face tutoring, etc. Yet these tools alone are not sufficient to ensure that the goal of student learning is being met. It remains for the educator to confirm that learning has taken place through some assessment of the knowledge acquired by the student.

2.2.1. Brief History of Educational Assessment

Today, privately grading some type of written work is the default approach for assessing student learning, but this is a relatively recent phenomenon which is not without controversy. From the time of the Scholastics in the late Medieval period through at least the 15th century (and generally until the mid-19th century), assessment of students was carried out orally, in public, and in Latin [93]. This consisted of a disputation, in which the degree candidate would orally present arguments in favor of some proposition while being challenged by an assigned adversary, the masters of the school, and any member of the academic community who happened to be in attendance. Over time, an increased Newtonian mathematical component in education, a rise in the number of students during the 19th century, and a change in the social and political environment of academia led to the decline of this system, first at Cambridge and then more widely, and to the normalization of an essay-based presentation of arguments which would remain common throughout the late 19th and 20th centuries.

At the turn of the twentieth century in America, the College Entrance Examination Board (later called the College Board) assessed college applicants using entirely essay-based exams which were manually composed and graded by teachers and professors. Within a few decades, the first multiple-choice tests were being designed, and by 1926, the multiple-choice Scholastic Aptitude Test (SAT) was released, though it did not gain dominance in college entrance examinations until the 1940s when World War II reduced the workforce available to grade entry exams. After the war, the Advanced Placement (AP) exams were designed with a substantial essay component. Over the past half century, standardized testing as a means to sort students (for college admission and employment purposes) and schools (for funding purposes) has skyrocketed but has remained a controversial feature in American education [67].

Since the advent of the Internet, distance-learning courses have begun to make up a greater and greater proportion of all post-secondary education. According to a recent study [3], over 6 million students in higher education were taking at least one online course during Fall 2010, which represents almost a third (31%) of the entire student body, nationwide. Since 2002, total enrollments have grown by 18% while the number of students taking online courses has out-paced this figure, growing by 283% over the same period with a growth of at least 10% each year. With the increasing availability of video lectures and other multimedia learning modules in these online courses, there exists an unprecedented opportunity for a practically limitless number of people to receive a quality education from a distance. Unfortunately, high-quality assessment remains a significant bottleneck for any large-scale distance-learning program.

The most common method of automating this assessment is by using simple assessment methods with clear right or wrong answers: e.g. multiple choice questions (MCQs), true-false questions, matching, etc.). These can all be reliably graded by a machine. Unfortunately, there are limitations associated with these types of questions, the most significant of which is that there is no way to determine partial understanding (and so provide partial credit). It is also much more difficult to measure a student's understanding of a concept beyond simple recognition and definitional familiarity [74]. One study published in 1980 [36] suggests that a reliance upon MCQs may also disadvantage students from a learning point of view. In this study students who had practiced using short answer questions were significantly better able to retain their knowledge for a short answer test than those who had practiced using only multiple-choice questions. Unlike multiple-choice questions, short answer responses and essay questions, train students to generate correct answers rather than to simply recognize them, and for these types of assessment items to be automated, an analysis of the texts themselves becomes necessary.

In a traditional assessment setting (e.g., an exam, assignment or quiz), an instructor or a grader is required to spend time providing students with feedback on their responses to questions related to the subject matter. In many cases, however, a competent instructor is not available to provide this feedback or is unable to handle the magnitude of the work required. This is where automation comes into play.

2.2.2. Computer-Aided Assessment

Computer-aided assessment has been used to reduce the burden on instructors since at least the mid-1960s when test grading machines, now ubiquitous in American schools and around the world, were first being employed to automate the marking of multiple choice assignments [10]. Around the same time, Ellis Page developed the influential Project Essay Grade (PEG) system [76] which sought to apply computational power to more complex essay-style problems.

In more recent years, intelligent tutoring systems (ITS) have come to be a rich avenue of CAA research due to the clear benefits of tutoring on student learning, the low-stakes context of grading and feedback in a tutoring scenario,and the significant time burden that tutoring places on instructors and other human tutors. Since the necessary level of one-on-one interaction is untenable for an individual instructor with more than a handful of students, tutoring is not applied as widely as it should perhaps be in the context of higher education. However, with the aid of computerized learning modules that assess a student's progress and give feedback, it is becoming increasingly possible to simulate one-on-one instruction in a way that is advantageous to student learning [6].

It has been reported that interactive tutoring with a human tutor produces advantages to student learning (over and above simple lecture-based learning), raising marks by up to 2 standard deviations [22]. At the same time, state-of-the-art intelligent tutoring systems have been reported to raise marks by 1 standard deviation (SD) [4]. However, more recent scholarship [100] has suggested that both of these reports are erroneous and that the actual deviation improvements are closer to 0.79 SDs for a human tutor and 0.76 SDs for an ITS tutor, which suggests that state-of-the-art ITS are now on par with human-level behavior.

I should also point out one complicating feature of the grading task which may be a source of reticence for students and instructors alike – namely the objectivity of the mechanized grader. This is a double-edged sword. On one hand, automating the process prevents unscrupulous graders from "playing favorites," but at the same time, this cold and mechanical form of grading goes against the inclination and preferences of students and teachers alike. While grading according to the recommended practices is meant to be used as feedback, purely indicating the quality of the work a student has produced, teachers are apt to assign grades as a reward, not for achievement, but for effort and to motivate students [15, 24]. For better or worse, this aspect of grading is lost in both this work and in CAA, generally.

2.3. Related Work

Research to date has concentrated on two subtasks of CAA: the grading of essays, which is done mainly by checking the style, grammaticality, and coherence of the essay (cf. [9, 44, 76]), and the assessment of short student answers (e.g., [80, 85, 94]). Each type of assessment can be either summative (geared towards producing a reliable and defensible grade in a high-stakes grading situation) or formative (used as a learning aid in a lower-stakes situation, such as a review session, self-check, or tutoring environment). The primary focus of this work is in automatic short answer grading with a lean towards summative assessment, though I believe that the methods described in Chapters 4 and 5 can be applied in either context.

2.3.1. Summative Assessment

Short answer grading systems that perform summative assessments in high-stakes situations must be impeccably accurate, so many state-of-the-art systems [70, 96] tend to place a lot of the burden on the instructors and the test designers themselves. Some attempt is usually made to, first, distill an answer into individual, correct answer components which must all be included for an student answer to be considered fully correct. Then, these answer components are manually crafted into models or patterns which must be matched by a student response to indicate that a given component has been successfully answered. These patters should account for differences in word choice, word order, passive or active voice etc.

The WebLAS system from UCLA [11], which can be considered more of an instructional aid than a grading system *per se*, parses a model answer in order to detect important components (e.g. words and phrases), supplements these using WordNet, and compiles a regular expression to match student answers. This is done interactively with the instructor, and has not – to my knowledge – been evaluated as a stand-alone grading system.

Another early foray into short answer grading [19], proposed modeling a correct answer as a set of Prolog conceptual dependencies, supplemented by a list of synonyms. It explicitly requires well-structured texts for student input (for the parser) and contains a separate syntax-analysis module which can be used as part of the grade. However, this system has never been formally evaluated.

If a large annotated corpus is available, the patterns to match can be supplemented by learning additional patterns semi-automatically. The Oxford-UCLES system [96] bootstraps patterns by starting with a set of keywords and synonyms and searching through windows of a text for new patterns. A later implementation of the Oxford-UCLES system [80] compares several machine learning techniques, including inductive logic programming, decision tree learning, and Bayesian learning, to the earlier pattern matching approach, with encouraging results.

CarmelTC [85, 101] treats grading as a text classification problem. The authors of the question produce a set of categories representing answers that are either correct, expressing the same idea in different ways, or indicate some misconception. The system then attempts to classify the student answers into one of the categories using a variety of approaches including latent semantic analysis (LSA), a naive Bayes (NB) classifier, and a decision-tree method based upon deep syntactical features on the student text. They report for their hybrid approach (combining the NB classifier with the decision tree) an F-measure of 0.85, well above their LSA and NB baselines (0.70 and 0.77, respectively). This evaluation was only performed on a single physics question with 126 student responses, so it is difficult to extrapolate their results to a more general context.

The Open University, a leader in distance learning in the U.K., recently began using assessment software commercially available from Intelligent Assessment Technologies (IAT) [51, 69, 70]. The system models correct answers as a set of keywords along with the role-based relationships between them. These are then abstracted to become an answer template by allowing for lexical variance among the keywords. Student answers can be flagged as close or requiring a certain type of feedback. They reported 96.6% agreement with human annotators for their evaluation set.

Finally, the Educational Testing Service (ETS) has produced several systems which have sought to tackle the task of short answer grading. In their early work [17], they used a concept grammar and concept lexicon to build appropriate patterns for their answers based upon a training set of 200 student answers. Faced with a true/false decision, this early system achieved 81% accuracy.

Several years later, ETS released C-Rater [55], which matches the syntactical features of a student response (i.e. subject, object, and verb) to that of a set of correct responses. The gold-standard model patterns are built semi-automatically, by first converting each answer into a set of one or more predicate-argument tuples. Each word is supplemented with contextually similar words. In a move reminiscent of my work described in Section 4.2.2, graded student responses can also be converted into tuple form and used to grade other student responses. On a large-scale assessment by the National Assessment of Education Progress agency, C-Rater reported accuracy between 81% and 90%.

Current work on C-Rater [94] treats the grading task more like a textual entailment task. As in prior work, model answers, based upon the analysis of 100-150 graded student answers are broken into concepts to look for in a correct answer. Each concept is represented by a set of sentences supplemented by a lexicon, and scoring is based upon the presence or absence of concepts. Breaking new ground, however, student answers are parsed in order to extract a predicate-argument structure which is then categorized as absent, present, or negated for each concept using a maximum entropy-based matching algorithm. Reported agreement (per concept-match) was 84.8% compared to an annotator agreement of 90.3%.

2.3.2. Formative Assessment

The Geometry Explanation Tutor [2] was an attempt to aid in student understanding by constructively offering hints as students attempted to explain their reasoning behind geometry-related answers. The authors built a hierarchy of incorrect (or semi-correct) answers that indicate what specific knowledge a student seems to have upon arriving at a given incorrect (or semi-correct) answer. The student can then be guided to a more correct answer. For instance, "angles are congruent" and "angles in a triangle are congruent" are both incorrect statements on their way to the correct statement "angles opposite congruent sides in an isosceles triangle are congruent." They have modeled their system as a classification task. Each node in the hierarchy has a set of texts to represent them. Student responses are parsed and fed to a Loom classifier which assigns it to some node in the hierarchy. They report an accuracy of 80%.

In the dependency-based classification component of the Intelligent Tutoring System [75], instructor answers are parsed, enhanced, and manually converted into a set of content-bearing dependency triples or facets. For each facet of the instructor answer each student's answer is labelled to indicate whether it has addressed that facet and whether or not the answer was contradictory. The system uses a decision tree trained on part-of-speech tags, dependency types, word count, and other features to attempt to learn how best to classify an answer/facet pair.

AutoTutor [39, 104] has been designed as an immersive tutoring environment with a graphical "talking head" and speech recognition to improve the overall experience for students. AutoTutor eschews the pattern-based approach favored by higher-stakes systems in favor of a bag-of-words (BOW) LSA approach. In addition to analyzing student responses to determine if they match a correct response, AutoTutor also makes pedagogical decisions regarding which type of response to give a student: hints, pumps for more information, correcting a student response, giving the answer directly, etc. In order to make these determinations, the authors analyzed a corpus containing 100 hours worth of human tutoring. In this corpus, 192 student answers were found, rated separately by four raters, and then used to evaluate the grading module of the AutoTutor system. It was found to correlate with the annotators at r=0.49 while a pair of humans with an intermediate knowledge of the subject correlated with one another at r=0.51. Interestingly, a pair of human experts correlated with one another at r=0.78.

An offshoot of AutoTutor, called Research Methods Tutor, has been involved in several studies [6, 103] of the effect that interactive tutoring systems (ITS) have on student learning compared to simpler learning environments where students read through a prepared text and answer a short multiple choice quiz at the end – called computer-aided instruction (CAI) systems. Students were given a pretest at the beginning of the semester and a post-test at the end, and were assigned to various tutoring systems. Results indicate that participating in the interactive tutoring system improved scores (above the simple act of taking the course) by 0.75 standard deviations (SDs). Human tutoring has been found in other work to improve scores by 2.3 SDs. It was found that ITS showed an improvement over CAI as well, raising scores by 13.5% compared to 8.8% for CAI.

2.3.3. Sentence Similarity

A natural starting point for research into sentence similarity is in the application of existing document similarity techniques to sentence similarity problems. However, even leaving aside the issue of syntax, the traditional vector-based measures of similarity are insufficient in themselves to adequately measure the similarity between short texts. With such a small number of words, there simply isn't enough context to reliably model the two texts as vectors of words. Of course, the text representations can be expanded in many ways.

Following the vector-space models used in document similarity (or query/document similarity) tasks, much research has been performed that involves using a corpus as a context within which individual words can be modeled as a vector in some multi-dimensional semantic space – thereby permitting a full sentence to be so modeled as the sum of its constituent word vectors. Three such techniques deserve special mention.

Latent semantic analysis [29, 53] has been widely used in recent decades to model texts. Briefly, a corpus is represented as an N by N matrix which contains information on the co-occurrence of word pairs within a window in some document collection. This matrix then undergoes singular value decomposition to reduce the dimensionality of the matrix to reduce sparseness and make the vector size computationally feasible. However, this causes the word co-occurrence information to be hidden, thus making the semantic knowledge it contains latent. A fairly robust analysis of LSA as it applies to document similarity was performed by Lee et al. [56] showing the effect of various models of similarity and of varying the parameters for training an LSA model.

Explicit semantic analysis [35], on the other hand, makes use of the high dimensional space directly – typically using Wikipedia articles as the dimensions and tracking term frequency within each article. Using the same Wikipedia article dimensionality, salient semantic analysis [41] uses term co-occurrence with the hyperlinks within Wikipedia to detect salient terms and to weight the concept vector in favor of these important (or more salient) dimensions for a given word.

Other work builds upon research in term similarity to produce a measure for sentence similarity that is based strictly upon the similarity between existing words. In earlier work from my research group [68] a set of well-known metrics modeling term similarity as the inverse of a distance within the WordNet hierarchy or basing it upon the information content of a common ancestor. Liu et al. [64] extends this idea slightly by adding word-order information. The use of a thesaurus (either WordNet or Roget's to define word similarity (and extrapolate a sentence similarity) has been used in several cases, and shows impressive results for the sentence similarity task [52, 98]. In a similar way, another group has built upon the idea of dynamic time warping (effectively an edit distance with costs defined using WordNet relations) to produce a similarity score for sentences [63].

One early example [42], involved training a classifier to detect similarity (or none) based upon simple features like word order, distance between words, number of matched words, and verbs with matching Levin classifications [58]. Interestingly, this can be considered an attempt to include structure and syntax in a similarity decision. Islam and Inkpen [47] have successfully built upon these ideas to model sentence similarity in terms of both structural features such as word order and the least common subsequence of the sentences and term similarity measures such as those described above. Likewise, Li et al. [59] have attempted to combine word order, distance and depth within an ontology, and frequency statistics from a corpus to measure similarity.

Testing on datasets for both term similarity and document similarity (though not for sentence similarity), Yeh et al. [106] introduces a method for detecting similarity based upon the well-known PageRank algorithm [14]. Building upon existing work in PageRank-based lexical similarity [46], they measured similarity in a three step process. First, an initial weighted teleport distribution was computed based upon the Wikipedia "neighborhood" of the terms in the text using a modified version of the explicit semantic analysis (ESA) algorithm. Then, a personalized PageRank was applied to the texts. Finally, the resultant distribution vectors were compared using standard vector similarity techniques.

Although the field as a whole is underdeveloped, detecting sentence-level similarity is an important part of many NLP tasks including query substitution, image retrieval, improved document retrieval, machine translation evaluation, and text summarization. It also has applications in three tasks related to, but distinct from, general sentence-similarity: textual entailment, paraphrase detection, and computer-aided assessment (CAA). While CAA is the general focus of this work (see Section 2.2.2), I also describe (in Chapter 6) a portability study, applying these techniques to the other two related tasks. To that end, I here introduce the tasks of textual entailment and paraphrase detection.

2.3.4. Textual Entailment

The textual entailment task encourages computers to make inferences based upon a short thesis (T) and to determine whether a separate hypothesis text (H) can reasonably be understood to follow from it. This is a subconscious process in humans, and so it is easy to overlook, but in the field of computer understanding, it is a gaping hole that must be filled by somehow modelling common sense and its use in informal logic.

In order to promote research in textual entailment, the PASCAL organization begin hosting annual challenges in 2005 that allowed researchers to evaluate and benchmark their new and existing textual entailment systems. These challenges were called the Recognizing Textual Entailment (RTE) challenge [25], and they have garnered many impressive submissions. Despite the generally poor results of the first challenge, significant annual improvements in the quality of submissions has been the norm over the years. Importantly for my purposes, their data (from multiple challenges) is now publicly available for the formal or informal evaluation of entailment systems.

The entailment-related works that are most similar to what is described in this thesis are the graph matching techniques proposed by [40] and [87]. Both input texts are converted into a graph by using the dependency relations obtained from a parser. Next, a matching score is calculated by combining separate vertex- and edge-matching scores. The vertex matching functions use word-level lexical and semantic features to determine the quality of the match while the edge matching functions take into account the types of relations and the difference in lengths between the aligned paths. A similar avenue of research that heavily considers structure utilizes tree kernels to measure the similarity between syntactic graphs [73].

Following the same line of work in the textual entailment world are the research teams at Stanford [20, 26, 65, 82], which experiment variously with using diverse knowledge sources, using a perceptron to learn alignment decisions, and exploiting natural logic. Their work differs from that of many others in the entailment field in that they devote significant attention to detecting contradictions in the texts, which may invalidate an other ways similar (or entailing) pair [28].

Another structure-based line of research was carried out using so-called "dependency tree skeletons" [102]. Briefly, all paths through a dependency tree that contain keywords (nouns) are considered spines, and anything not attached to a spine are removed. Common prefixes or suffixes are removed and anything that remains is considered a mismatch between the two texts. Four kernels are combined to compute an overall classification based upon verb matches, verb relations, subsequence scores, and collocation scores.

Two other groups that are notable in the context of textual entailment are both associated with Language Computer Corporation. Hickl et al. [43] convincingly dominated the 3rd RTE challenge by using a 4-stage pipeline that first involves heuristically extracting a set of assertions from both texts. These assertions are aligned based upon a set of term-level lexical and semantic features with weights learned on the training data. After ranking and scoring the thesis assertions for each hypothesis assertion, the best hypothesis assertion is selected and classified as either entailing or non-entailing using a decision tree. Finally, the pairs are checked for contradictions which invalidate the entailment decision. The COGEX system [97] converted the T and H snippets into a "three-layered semantically-rich logic form" representation which was relaxed until reaching a threshold or detecting entailment. A set of named-entity heuristics were also included to eliminate false positives.

2.3.5. Paraphrase Detection

Paraphrase is a natural part of human communication. As we relate a story or anecdote that we have heard from someone else, we are almost guaranteed to paraphrase and reword the story rather than to engage in a rote recitation of the tale using the exact words with which it was told to us. This is not a natural trait of computers and so computational tasks that involve mimicking or understanding typical human speech require a certain comfort with paraphrase. In language generation for conversational agents, for instance, the ability to alter texts through paraphrase prevents the text from becoming stale and predictable. Likewise, in question answering, the ability to detect whether two answers are paraphrases of one another may serve to strengthen the case for the answer found.

In recent literature, the paraphrase task has been often been treated as a special case of two other tasks – textual entailment and machine translation. If textual entailment requires that H can reasonably be inferred from T, then detecting a paraphrase requires that both H and T can be inferred from one another [5, 88]. Similarly, paraphrase can be thought of as a case of monolingual translation, where the two texts convey the same information using different surface forms.

Finch et al. [32] make use of multiple tools in the machine translation evaluation arsenal. Simpler evaluation scores such as word error rate (WER), position-independent word-error rate (PER), the Bilingual Evaluation Understudy (BLEU) metric and the metric put forth by the National Institute of Standards and Technology (NIST) were used in conjunction with part-of-speech information and a WordNet-based measure of term similarity [48] to train a support vector machine (SVM) classifier to make a paraphrase decision.

Turning the paraphrase detection task around, Qiu et al. [81] attempts to detect semantically important dissimilarities between the two texts an in the absence of such dissimilarities claims to detect paraphrase. Constituent components of each text were broken down into predicate-argument structures. Non-matching structures were then classified as either significant or insignificant using an SVM classifier.

CHAPTER 3

DATASETS AND EVALUATION METRICS

Until recently, the computer-aided assessment (CAA) community has been characterized by isolated progress with little ability to compare the approaches of different groups and to build upon the work of other researchers. Without a publicly available dataset freed from legal issues associated with privacy and intellectual property rights, it was not possible to effectively compare two systems side-by-side. In order to address this anomaly (and to evaluate my grading methodology), I have created and publicized a dataset[1] consisting of short answer questions taken from introductory computer science assignments with answers provided by a class of undergraduate students. The assignments were administered as part of a course on Data Structures at the University of North Texas in the fall of 2007. For each assignment, the student answers were collected via an online learning environment.

3.1. Description of Short Answer Grading Dataset

The students submitted answers to 87 questions spread across ten assignments and two examinations. Six of these questions (4.6, 4.7, 8.5, 9.5, 9.7, and 12.3) were ignored over the course of these experiments as the question types were more similar to multiple choice,

[1]In cooperation with my advisor, Dr. Rada Mihalcea

TABLE 3.1. Two sample questions along with student answers and the grades assigned by the two human judges. In these examples, the annotator scores are reasonably close.

	Sample questions, correct answers, and student answers	Grades
Question:	What is the role of a prototype program in problem solving?	
Correct answer:	To simulate the behavior of portions of the desired software product.	
Student answer 1:	A prototype program is used in problem solving to collect data for the problem.	1, 2
Student answer 2:	It simulates the behavior of portions of the desired software product.	5, 5
Student answer 3:	To find problem and errors in a program before it is finalized.	2, 2
Question:	What are the main advantages associated with object-oriented programming?	
Correct answer:	Abstraction and reusability.	
Student answer 1:	They make it easier to reuse and adapt previously written code and they separate complex programs into smaller, easier to understand classes.	5, 4
Student answer 2:	Object oriented programming allows programmers to use an object with classes that can be changed and manipulated while not affecting the entire object at once.	1, 1
Student answer 3:	Reusable components, Extensibility, Maintainability, it reduces large problems into smaller more manageable problems.	4, 4

true/false, or ordering problems. For example, question 4.7 is worded as follows: "Using an index outside the bounds of the array generates an error. Is this a compilation error or a run-time error?"

Table 3.1 shows two question-answer pairs with three sample student answers each. Thirty-one students were enrolled in the class and submitted answers to these assignments. The data set used in this work consists of a total of 2273 student answers. This is less than the expected $31 \times 81 = 2511$ as some students failed to submit a few assignments.

This dataset has been released in two stages. The first release consisted of 21 questions (from assignments 1-3) with a total of 630 student responses. This dataset was released as part of my first publication related to this task [72] and is referred to as MM2009 for the remainder of this thesis. The full dataset was released (all 87 questions) along with my second publication [71]. This full dataset is referred to as MM2011.

For all experiments on the full dataset (MM2011), thirty-two student answers were used for development and were not included in the evaluation. More details on this process can be found in Section 5.3.

3.1.1. Annotation Process

The answers were independently graded by two human judges, using an integer scale from 0 (completely incorrect) to 5 (perfect answer). Both human judges were graduate students in the computer science department; one (Grader 1) was the teaching assistant assigned to the Data Structures class, while the other (Grader 2) was myself performed after the course had ended. The average grade of the two annotators is treated as the gold standard against which the system output is compared.

TABLE 3.2. Magnitude of difference between annotators

Difference	Examples	% of examples
0	1294	57.7%
1	514	22.9%
2	231	10.3%
3	123	5.5%
4	70	3.1%
5	9	0.4%

The annotators were given no explicit instructions on how to assign grades other than the [0..5] scale. Both annotators gave the same grade 57.7% of the time and gave a grade only 1 point apart 22.9% of the time. While the agreement between annotators was 57.7% and the adjacent agreement (off by at most one) was 80.6%, the Kappa statistic was 0.28 indicating only moderate agreement excluding chance. This is likely due to the high proportion of answers (for both annotators) that were judged perfectly correct (i.e. 5). Comparing each annotator to the average, their root mean square error was 0.66. The annotators correlated (using Pearson's r) at 0.59.

A full breakdown of the divergence in annotator grading can be seen in Table 3.2. A sample of responses in which the annotators differed by more than 1 can be found in Table 3.3.

TABLE 3.3. Two sample questions along with student answers and the grades assigned by the two human judges

	Sample questions, correct answers, and student answers	Grades
Question:	What is the scope of global variables?	
Correct answer:	File scope.	
Student answer 1:	they can be accessed by any C++ file anywhere.	5, 0
Student answer 2:	Global Variables can be used in any function as long as the appropriate .h file that holds the variable is included	5, 3
Student answer 3:	can be accesed by any classes that have and object of that variables class in it	5, 1
Question:	What is a stack?	
Correct answer:	A data structure that can store elements, which has the property that the last item added will be the first to be removed (or last-in-first-out)	
Student answer 1:	Stores a set of elements in a particular order.	4, 0
Student answer 2:	A stack is an ADT that stores a set of elements in a particular order.	4, 1
Student answer 3:	a finite ordered list with zero or more elements	4, 0

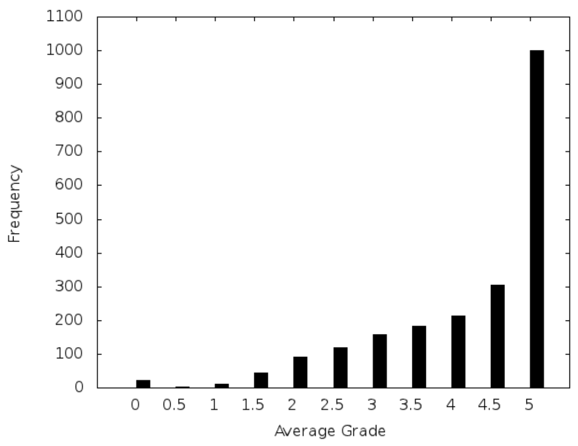
In addition, an analysis of the grading patterns indicate that the two graders may have been operating off of different internal grading policies as one grader (Grader 1) was considerably more generous than the other. In fact, when the two differed, Grader 1 gave the higher grade 76.6% of the time. The average grade given by Grader 1 is 4.43, while the average grade given by Grader 2 is 3.94. Details can be found in Table 3.4.

For both annotators, and hence for the gold standard mean, the dataset is heavily biased towards correct answers (see Figure 3.1). However, I believe that this correctly mirrors real-world issues associated with the task of grading.

TABLE 3.4. Annotation confusion matrix: Columns indicate the grade given by Grader 1 while rows indicate the grade given by Grader 2

	0	1	2	3	4	5
0	22	3	4	33	36	9
1	0	6	5	37	50	34
2	1	5	15	56	58	37
3	0	2	3	61	110	70
4	0	0	4	30	86	183
5	0	0	3	55	119	1104

FIGURE 3.1. Real-world grading biased towards correct answers



3.2. Other Datasets

In order to evaluate the applicability of the techniques used in this system to other related fields, I have made use of an additional five datasets that have been widely used in the fields of text similarity, textual entailment, and paraphrase detection.

The first dataset, hereafter referred to as the SemEval2012 corpus, was provided as part of the Semantic Evaluation (SemEval) 2012 workshop for semantic textual similarity (STS) [1]. This dataset was released in two stages (training and testing). The training data was composed of data from 3 existing datasets. Formally, 750 sentence pairs were taken from the Microsoft Research Paraphrase Corpus [MSRpar] (see below), 750 pairs were taken from the Microsoft Research Video Description Corpus [MSRvid], and 734 pairs were taken from the 2008 Workshop on Machine Translation (WMT) development dataset [SMTeuroparl] (Europarl section). The testing data that was released was taken from these same three datasets (750 [MSRpar], 750 [MSRvid], 459 [SMTeuroparl]) as well as from two datasets which were a surprise to the entrants. The first of these two datasets [OnWN] consisted of 750 pairs where the first sentence was taken from OntoNotes and the second was taken from a WordNet definition. In the second [SMTnews], 399 pairs were extracted from the news conversation section of WMT. Altogether, this represents 2234 training pairs and 3108 testing pairs – or 5342 pairs combined. Each of these pairs was given a score on a scale of [0..5] by five users associated with Amazon Mechanical Turk. The average of the users' scores is treated as the gold standard similarity score. Results from experiments on this dataset can be found in Section 6.1.

The second dataset, hereafter referred to as Li30 [60], was based on a dataset built by Rubenstein and Goodenough [86] in 1965 to detect similarity between individual terms. The new dataset was formed by replacing the individual terms with their definitions taken from the Collins Cobuild [91]. New similarity scores were calculated by taking the average similarity (on a [0..4] scale) as judged by 32 native English speakers. In order to reduce the effects of bias on the dataset, a subset of 30 pairs is commonly used in research on this dataset, and I follow this norm in this set of experiments. Results from experiments on this dataset can be found in Section 6.2.1.

The third dataset, hereafter referred to as Lee50 [56], is a collection of 50 small documents (between 51 and 126 words) each taken from the Australian Broadcasting Corporation's news mail service. All possible pairs of these documents (i.e. 1275 pairs) were given to a grouping of 83 college students who rated the similarity of the document pairs on a scale of [1..5]. The dataset was heavily skewed towards low similarity scores. An analysis of the annotations revealed an average inter-rater correlation of 0.605 by repeatedly selecting a random set of annotations and comparing to the average of the remaining annotations. It was reported that around 90% of annotations were within 1 point of the average. The average of the annotations for each pair were scaled to the [0..1] range and treated as a gold-standard for similarity. Results from experiments on this dataset can be found in Section 6.2.2.

The fourth dataset, hereafter referred to as RTE-3 is from the third Recognizing

Textual Entailment (RTE) Challenge organized by the PASCAL group [25]. For system development, 800 sentence pairs were provided, and for testing, another 800 sentence pairs. These sentence pairs were each manually tagged as either "entailing" or "non-entailing." The sentence pairs were drawn from four broadly defined categories: information retrieval, multi-document summarization, information extraction, and question answering[2] Twenty-six teams participated in the RTE-3 challenge [37]. Results from experiments on this dataset can be found in Section 6.2.3.

The final dataset, the Microsoft Research Paraphrase (MSRP) corpus, was published by Microsoft's Natural Language Processing Group [30]. Using heuristic extraction techniques and a support vector machine (SVM) classifier, 5801 sentence pairs were removed from a collection of news articles gathered from the World Wide Web over the course of 2 years. All candidate sentences were required (by the extraction heuristics) to be similar in length, to have at least three words in common, and to have a moderate to large edit distance (greater than 7 edits). Two independent judges annotated each pair as either "semantically equivalent" or not with ties broken by a third judge. Overall, 67% of the pairs were found to be equivalent. Results from experiments on this dataset can be found in Section 6.2.4.

3.3. Evaluation Metrics

In attempting to analyze the results of this work, I have been faced with the difficult decision of which metric to use to evaluate the scores provided through a comparison with the gold standard. Related work has been split on which evaluation metric is the most appropriate, with the decision often determined by the dataset.

For datasets with real-valued scores, such as SemEval2012, Li30 and Lee50 [56], correlation metrics such as Pearson's r and Spearman's ρ are more frequently used [35, 39, 41, 98]. For datasets such as the RTE suite or the MSRP Corpus that require a binary yes-no decision or assignment to a category, measures such as accuracy, Kappa statistic, and precision/recall are more common [18, 62, 94, 95]. I believe that both measures are inherently limited and

[2]See the RTE-3 website (http://pascallin.ecs.soton.ac.uk/Challenges/RTE3/Introduction/) for further details on the creation process.

misleading when used alone on a short answer grading set.

For instance, the correlation statistics are undefined in the case that every student gets the same grade on a given problem. This occurs several times in this dataset. In addition, the change from 6 possible grades that the two annotators provide to 11 possible grades that the average score can take makes comparison between the annotators' correlation and any system's correlation much less meaningful. Likewise, for accuracy, Kappa, and precision/recall measures large grading discrepancies and small grading discrepancies are penalized to the same extent. In reality, a system that awards an "A" effort with a "B" is more satisfactory than a system that awards an "F" for the same effort. Perhaps more problematic is that any metric that requires an exact match such as accuracy and the Kappa statistic require real values to be "rounded" in order to match a category.

In my earlier efforts (See Chapter 4), I reported only Pearson's r, but as the work matured, I have chosen to report also the root mean squared error (RMSE) to quantify the difference between the system response and the given scores. Where accuracy scores are reported for real-valued datasets, it should be assumed that scores have first been rounded to the nearest 0.5.

CHAPTER 4

BAG-OF-WORDS APPROACHES

In this chapter, I report my early work in the application of existing knowledge-based and corpus-based similarity measures to the task of automatic short answer grading. In particular, I am interested in determining how the size and subject matter of the training corpora affect the overall quality of the corpus-based measures. Furthermore, I describe an attempt to enhance the provided instructor answer (to account for phrasal variation) by using other student responses in a manner similar to the pseudo-relevance feedback technique commonly employed in information retrieval. All experiments reported in this chapter were evaluated using the MM2009 dataset unless otherwise indicated.

4.1. Textual Similarity Measures

Comparative evaluations were performed using eight knowledge-based measures of semantic similarity (shortest path, Leacock & Chodorow, Lesk, Wu & Palmer, Resnik, Lin, Jiang & Conrath, Hirst & St. Onge), and three corpus-based measures (cosine similarity, latent semantic analysis, and explicit semantic analysis).

For the knowledge-based measures, I derive a text-to-text similarity metric by using the methodology proposed in [68]: for each open-class word in one of the input texts, the maximum semantic similarity that can be obtained by pairing it up with individual open-class words in the second input text is used. More formally, for each open-class word W of class C in the instructor answer, find maxsim(W,C) where

$$maxsim(W, C) = \max Sim_x(W, w_i)$$

where w_i is a word in the student answer of class C and the Sim_x function is one of the functions described below. The lexical similarity scores are determined using the WordNet::Similarity package described in [79]. All the word-to-word similarity scores obtained in this way are summed up and normalized to account for the length of the two input texts. A short description of each of these similarity metrics is provided below.

4.1.1. Knowledge-Based Measures

The shortest path similarity is determined as:

(1) $$Sim_{path} = \frac{1}{length}$$

where *length* is the length of the shortest path between two concepts using node-counting (including the end nodes).

The Leacock & Chodorow (LCH) [54] similarity is determined as:

(2) $$Sim_{lch} = -\log \frac{length}{2 * D}$$

where *length* is the length of the shortest path between two concepts using node-counting, and D is the maximum depth of the taxonomy.

The Lesk similarity of two concepts is defined as a function of the overlap between the corresponding definitions, as provided by a dictionary. It is based on an algorithm proposed by Lesk [57] as a solution for word sense disambiguation.

The Wu & Palmer (WUP) [105] similarity metric measures the depth of two given concepts in the WordNet taxonomy, and the depth of the least common subsumer (LCS), and combines these figures into a similarity score:

(3) $$Sim_{wup} = \frac{2 * depth(LCS)}{depth(concept_1) + depth(concept_2)}$$

The measure introduced by Resnik [83] returns the information content (IC) of the LCS of two concepts:

(4) $$Sim_{res} = IC(LCS)$$

where IC is defined as:

(5) $$IC(c) = -\log P(c)$$

and $P(c)$ is the probability of encountering an instance of concept c in a large corpus.

The measure introduced by Lin [61] builds on Resnik's measure of similarity, and adds a normalization factor consisting of the information content of the two input concepts:

(6)
$$Sim_{lin} = \frac{2 * IC(LCS)}{IC(concept_1) + IC(concept_2)}$$

I also consider the Jiang & Conrath (JCN) [48] measure of similarity:

(7)
$$Sim_{jcn} = \frac{1}{IC(concept_1) + IC(concept_2) - 2 * IC(LCS)}$$

Finally, the Hirst & St. Onge (HSO) [45] measure of similarity is considered, which determines the similarity strength of a pair of synsets by detecting lexical chains between the pair in a text using the WordNet hierarchy.

4.1.2. Corpus-Based Measures

The corpus-based measures differ from knowledge-based methods in that they do not require any encoded understanding of either the vocabulary or the grammar of a text's language. A semantic model is formed by analyzing alternatively the frequency of a word in the corpus, the words that it appears with, or the set of documents that it appears in.

In many of the real-world scenarios where computer-aided assessment (CAA) might be advantageous, robust language-specific resources (e.g. WordNet, dependency parsers, part-of-speech taggers) may not be available. Thus, state-of-the-art corpus-based measures may be the only available approach to CAA in languages with scarce resources.

One of the oldest corpus-based measures of document similarity is a vector-based cosine similarity using term frequency and document frequency (referred to as "tf*idf") [50]. Briefly, tf*idf works by converting a text (usually a document) into a vector representation, where each dimension of the vector is a unique word in a collection of documents. The values associated with each element of the vector are calculated as the product of the term frequency (tf) – the number of occurrences of this word in the document – and the inverse document frequency (idf) – the log of the ratio of documents in the collection to documents that contain this word. The vectors are then compared by finding the angle between them (cosine similarity). This technique is sometimes altered using novel smoothing or weighting policies, but the method as described above is used here, as a baseline. Document frequency is taken from the British National Corpus (BNC) [8].

Over the past two decades, latent semantic analysis (LSA), proposed by Deerwester et al. [29], has been widely used as a measure of similarity and has even been proposed as an all-inclusive model for human cognition of language [53]. In LSA, term co-occurrences in a corpus are captured by means of a dimensionality reduction performed by a singular value decomposition (SVD) on the term-by-document matrix $\mathbf{T}$ representing the corpus. The result is an independent vector for each word, which can be summed to form a new vector representing the full text. Again, the cosine similarity of the vectors is used to compute a text similarity score. For the experiments reported involving the MM2009 corpus, the SVD operation has been run on several corpora including both the BNC (LSA BNC) and a dump of the entire English Wikipedia (LSA Wikipedia). For the experiments reported in this chapter, the Wikipedia corpus refers to a version downloaded in September 2007.

Explicit semantic analysis (ESA) [35] is a variation on the standard vectorial model in which each dimension of the vector is directly equivalent to an abstract concept. Each article in Wikipedia represents a concept in the ESA vector. The relatedness of a term to a concept is defined as the tf*idf score for the term within the Wikipedia article, and the relatedness between two words is the cosine of the two concept vectors in a high-dimensional space. I refer to this method as ESA Wikipedia.

4.2. Experimental Setup

For the knowledge-based measures, I use the WordNet-based implementation of the word-to-word similarity metrics, as available in the WordNet::Similarity package [77]. For the LSA experiments, I use the InfoMap package.[1] For the experiments using explicit semantic analysis, I use an in-house implementation of the ESA algorithm as described in [34][2]. Note that all the word similarity measures are normalized so that they fall within a [0..1] range.

[1]http://infomap-nlp.sourceforge.net/

[2]Originally implemented by Samar Hassan with my own modifications to improve efficiency

The normalization is performed by dividing the original similarity score by the maximum possible score for that measure.[3].

4.2.1. The Role of Domain and Size

One of the key considerations when applying corpus-based techniques is the extent to which the size and subject matter of the training corpus affect the overall performance of the system. In particular, based on the underlying processes involved, the LSA and ESA corpus-based methods are expected to be especially sensitive to changes in domain and size. The language models that are built depend upon the relatedness of the words in the training data which suggests that, for instance, in a computer science domain the terms "object" and "oriented" will be more closely related than in a general-purpose text. Similarly, a large amount of training data will lead to less sparse vector spaces, which in turn is expected to affect the performance of the corpus-based methods.

With this in mind, two training corpora were developed for use with the corpus-based measures that sought to cover the computer science domain. The first corpus (LSA slides) consists of several online lecture notes associated with the class textbook, specifically covering topics that are used as questions in the dataset. The second domain-specific corpus is a subset of the Wikipedia dump (LSA Wikipedia CS) consisting of articles that contain any of the following terms: computer, computing, computation, algorithm, algorithms, recursive, or recursion.

The performance of the LSA models that have been trained on the domain-specific corpora is compared with LSA models trained on the open-domain corpora mentioned in Section 4.1.2, namely LSA Wikipedia and ESA Wikipedia. In addition, for the purpose of running a comparison with the LSA slides corpus, I also created a random subset of the LSA Wikipedia corpus approximately matching the size of the LSA slides corpus. I refer to this corpus as LSA Wikipedia (small).

[3]For several similarity measures this was done incorrectly and has been fixed for all experiments with the MM2011 dataset as shown in Chapter 5. The original published results are reported here.

Table 4.1 shows an overview of the various corpora used as training in these experiments, along with the Pearson correlation coefficient observed for this dataset.

TABLE 4.1. Corpus-based measures trained on corpora from different domains and of different sizes – MM2009 dataset

Measure - Corpus	Size	Correlation
Training on generic corpora		
LSA BNC	566.7MB	0.4071
LSA Wikipedia	1.8GB	0.4286
LSA Wikipedia (small)	0.3MB	0.3518
ESA Wikipedia	1.8GB	0.4681
Training on domain-specific corpora		
LSA Wikipedia CS	77.1MB	0.4628
LSA slides	0.3MB	0.4146
ESA Wikipedia CS	77.1MB	0.4385

Assuming a corpus of comparable size, it is expected that a measure trained on a domain-specific corpus would outperform a measure trained on a generic one. Indeed, by comparing the results obtained with LSA slides to those obtained with LSA Wikipedia (small), it is possible to see that by using the in-domain computer science slides the system obtains a correlation of $r=0.4146$, which is higher than the correlation of $r=0.3518$ obtained with a corpus of the same size but open-domain. The effect of the domain is even more pronounced when the performance obtained with LSA Wikipedia CS ($r=0.4628$) is compared with the one obtained with the full LSA Wikipedia ($r=0.4286$).[4] The smaller, domain-specific corpus performs better, despite the fact that the generic corpus is 23 times larger and is a superset of the smaller corpus! This suggests that for LSA the quality of the texts is vastly more important than the quantity.

When using the domain-specific subset of Wikipedia, decreased performance is observed with ESA compared to the full Wikipedia space. I suggest that for ESA the high-dimensionality of the concept space[5] is paramount, since many relations between generic words may be lost to ESA that can be detected latently using LSA.

[4]The difference was found significant using a paired t-test ($p<0.001$).

[5]In ESA, all the articles in Wikipedia are used as dimensions, which leads to about 1.75 million dimensions in the ESA Wikipedia corpus, compared to only 55,000 dimensions in the ESA Wikipedia CS corpus.

In tandem with my exploration of the effects of domain-specific data, I also look at the effect of size on the overall performance. The main intuitive trends are there, i.e. the performance obtained with the large LSA-Wikipedia is better than the one that can be obtained with LSA Wikipedia (small). Similarly, in the domain-specific space, the LSA Wikipedia CS corpus leads to better performance than the smaller LSA slides data set. However, an analysis carried out at a finer-grained scale, in which the performance obtained with LSA is calculated when trained on 5%, 10%, ..., 100% fractions of the full LSA Wikipedia corpus, did not reveal a close correlation between size and performance, which suggests that further analysis is needed to determine the precise effect of corpus size on performance.

4.2.2. Pseudo-Relevance Feedback

When a grader determines that a student answer is correct, it implies that there is some degree of similarity between the answer provided by the student and some correct answer provided by the instructor (or otherwise known to the grader). Since, in the interest of simplicity, the system is provided with only one correct answer, some student answers may be wrongly graded because of little or no similarity to the surface forms of that single correct answer.

In order to address this problem, I introduce a novel technique that feeds back from the student answers themselves in a manner similar to the way pseudo-relevance feedback is used in information retrieval [84]. In so doing, the paraphrasing that is usually observed across student answers will enhance the vocabulary of the correct answer, while at the same time maintaining the correctness of the gold-standard answer. In fact, something similar is done manually in most state-of-the-art pattern matching systems for CAA [17, 18, 95]. In these, a set of 100-400 student answers are analyzed and used to create patterns to overcome variety in word choice. I attempt to achieve the same goal using a simpler version of this system to automatically find correct answers among student answers.

Briefly, given a metric that provides similarity scores between the student answers and the correct answer, scores are ranked from most similar to least. The words of the top N ranked answers are then added to the gold standard bag-of-words. The remaining answers are then rescored according the the new gold standard vector. In practice, the scores from the first run (i.e. with no feedback) are held constant for the top N highest-scoring answers, and the second-run scores for the remaining answers are multiplied by the first-run score of the Nth highest-scoring answer. In this way, the original scores for the top N highest-scoring answers are kept (and thus prevent them from becoming artificially high), which at the same time, guarantees that none of the lower-scored answers will get a new score higher than the best answers.

The effects of relevance feedback are shown in Figure 4.1, which plots the Pearson correlation between automatic and human grading (Y axis) versus the number of student answers that are used for feedback (X axis). This experiment has not yet been attempted with the larger MM2011 dataset.

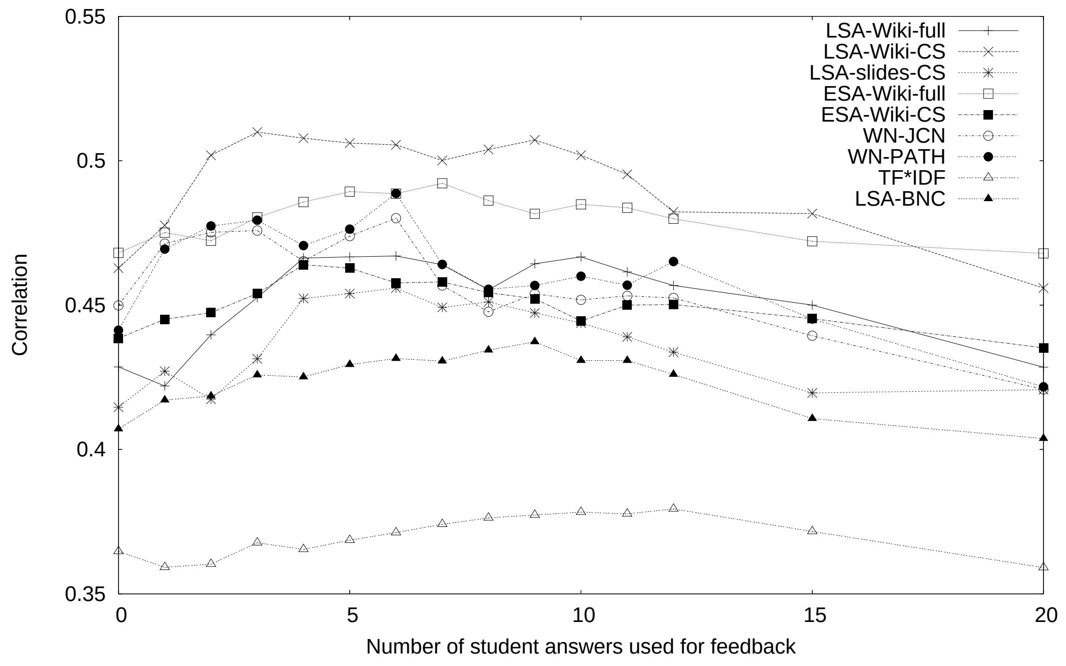
TABLE 4.2. Maximum absolute improvement obtained with relevance feedback for different measures – MM2009 dataset

Measure	Feedback answers	Maximum improvement
LSA-Wiki-full	6	0.0384
LSA-Wiki-CS	3	0.0471
LSA-slides-CS	6	0.0413
ESA-Wiki-full	7	0.0241
ESA-Wiki-CS	7	0.0255
WordNet-JCN	6	0.0302
WordNet-Path	6	0.0474
tf*idf	12	0.0147
LSA-BNC	9	0.0302

Table 4.2 shows the maximum absolute improvement that can be obtained by using feedback with a selection of measures, along with the number of feedback answers for which this improvement is obtained.

Overall, an improvement of up to 0.047 on the [0..1] Pearson scale is shown to be obtainable by using this technique, with a maximum improvement observed after about 4-6 iterations on average. After an initial number of high-scored answers, it is likely that the correctness of the answers degrades, and thus the decrease in performance observed after an initial number of iterations. These results indicate that the LSA and WordNet similarity metrics respond more favorably to feedback than the ESA metric. It is possible that supplementing the bag-of-words in ESA (with e.g. synonyms and phrasal differences) does not drastically alter the resultant concept vector, and thus the overall effect is smaller.

FIGURE 4.1. Effect of relevance feedback on performance – MM2009 dataset



4.2.3. Binary Decision

To gain further insights, I performed an additional experiment to determine the ability of this system to make a binary accept/reject decision. In this evaluation, the [0..5] human grading scale of the dataset is mapped to an accept/reject annotation by using a threshold of 2.5. Every answer with a grade higher than 2.5 is labeled as "accept," while every answer below 2.5 is labeled as "reject." Next, I use the best system (LSA trained on domain-specific data with relevance feedback), and run a ten-fold cross-validation on the data set. Specifically, for each fold, the system uses the remaining nine folds to automatically identify a threshold to maximize the matching with the gold standard. The threshold identified in this way is used to automatically annotate the test fold with "accept"/"reject" labels.

The ten-fold cross validation resulted in an accuracy of 92%, indicating the ability of the system to automatically make a binary accept/reject decision. However, moving the threshold to 4.0 (to distinguish between excellent and good answers) proved a more difficult task. Note that it is possible that the high-score bias associated with this dataset (see Chapter 3) is responsible for the high accuracy in this section.

4.3. Discussion

These experiments show that several knowledge-based and corpus-based measures of similarity perform comparably when used for the task of short answer grading. However, since the corpus-based measures can be improved by accounting for domain and corpus size, the highest performance can be obtained with a corpus-based measure (LSA) trained on a domain-specific corpus. Further improvements were also obtained by integrating the highest-scored student answers through a relevance feedback technique.

Table 4.3 summarizes the results of my experiments. In addition to the per-question evaluations that have been reported throughout this thesis, a per-assignment evaluation is also reported, which reflects a cumulative score for a student on a single assignment, as described in Chapter 3.

TABLE 4.3. Summary of results obtained with various similarity measures, with relevance feedback based on six student answers. Also listed, as baselines, are tf*idf and LSA trained using BNC. The annotator agreement (Pearson's r) is also shown.

	Correlation	
Measure	per-quest.	per-assign.
Baselines		
tf*idf	0.3647	0.4897
LSA BNC	0.4071	0.6465
Relevance Feedback based on Student Answers		
WordNet shortest path	0.4887	0.6344
LSA Wikipedia CS	0.5099	0.6735
ESA Wikipedia full	0.4893	0.6498
Annotator agreement	0.6443	0.7228

Overall, in both the per-question and per-assignment evaluations, the best performance was obtained by using an LSA measure trained on a medium size domain-specific corpus obtained from Wikipedia, with relevance feedback from the four highest-scoring student answers. This method improves significantly over the tf*idf baseline and also over the LSA trained on BNC model, which has been used extensively in previous work. The differences were found to be significant using a paired t-test ($p<0.001$).

In the next chapter, I use the knowledge-based and corpus-based techniques described here in the development of a system that takes into account not only the lexical-semantics of a sentence, but the structure as well through the use of dependency graph alignments. The scores of these simpler similarity measures are also used as features in training a support vector machine (SVM) learning system.

CHAPTER 5

ALIGNMENT SYSTEM AND SVM LEARNING

In this chapter, I first revisit the bag-of-words approaches discussed in the previous chapter and introduce some common-sense modifications to the preprocessing and evaluation processes. Afterwards, I detail my work using dependency graph alignment scores to account (in a simple way) for structural similarities between student and instructor answers. I go on to describe my efforts to exploit support vector machine (SVM) learning techniques to leverage non-redundant evidence spread across multiple simpler features. All experiments described in this chapter use the MM2011 dataset unless otherwise indicated.

It was shown in Chapter 4 and in much prior work that simple bag-of-words similarity metrics perform reasonably well when used in computer-aided assessment (CAA) and when detecting textual similarity in general. Unfortunately, by their very nature, fine differences in a word's role in the sentence narrative are incapable of being distinguished by these simple measures alone. Any student of poetry is aware that the sentence "Dido loves Aeneas" does not mean the same thing as "Aeneas loves Dido," but latent semantic analysis (LSA) and the other bag-of-words similarity measures are forever committed to a belief in requited love.

When grading student answers, this ability to detect these subtle differences in word order and sentence role are often the difference between a correct answer and an incorrect one. Most prior attempts to account for this in CAA have focused on using a large development set of student answers (usually over 100 answers per question) to manually craft patterns with a fixed word order which must be matched in order for a positive score to be given. This was in no way an option as the dataset under consideration has only around 30 student responses per question altogether.

In the textual entailment field, where such differences are the rule rather than the exception, much recent work [65, 78, 87] has focused on comparing the dependency graph representations of both texts in order to determine whether the hypothesis is entailed based upon the structure of the sentences without being tied to an explicit word order. These systems produce an alignment based upon the separate similarities of the nodes (words) and the edges (relations) of the dependency graphs and use the alignment itself to reach a decision regarding entailment.

In order to meet the needs of the grading task, my system attempts to quantify an alignment as a score on a [0..5] scale so that it can be directly applied as a grade for a student answer. As an initial attempt, I have modeled the dependency graph alignment step as an assignment problem and have sought to match nodes in such a way as to maximize the alignment score (i.e. the grade). This process is described in more detail in Section 5.4. Before describing the three-staged pipeline system (as shown in Figure 5.1), I must first discuss a few pragmatic changes that are used throughout this chapter.

5.1. Modifications and Bag-of-Words Improvements

Before looking at the system pipeline, I describe here two techniques that deserve mention: *question demoting* and *isotonic regression*. The former is based upon real-world intuition among those who frequently grade student answers – namely that repeating words in the question is easy and is not necessarily indicative of student understanding. For example, given the question âĂİWho was the first President of the United States?âĂİ and an instructor answer âĂİThe first president of the United States was George WashingtonâĂİ, it may be possible for a student to receive a fair amount credit (due to surface similarities) by providing the answer âĂİThe first president of the United States was Barack Obama.âĂİ In order to prevent students from inappropriately receiving credit for these types of answers, terms that are freely given to the student in the question are discounted. Specifically, any terms contained in the question are removed from both the instructor answer and the student answer. This makes it possible to compare only the pertinent terms. I have termed this process question demoting. Coincidentally, I have recently discovered that this method was quietly applied in an earlier work on CAA [96], but has never been analyzed in any detail.

The second technique to mention here – isotonic regression – is applied to meet an evaluation-related need. I have discussed in Section 3.3 the decision to report results measuring correlation, accuracy, and root mean squared error (RMSE). In order to determine accuracy and RMSE, it is necessary for the system output and the gold-standard grades to be on the same [0..5] scale. Additionally, it is vital that a grading system be able to give students grades that have some inherent or agreed upon meaning – e.g. the standard 100 point scale. With this in mind, I use isotonic regression [107] to convert the system scores onto the same [0..5] scale used by the annotators[1]. The isotonic regression model is trained on each type of system output (i.e. BOW scores, alignment scores, SVM output). More details on the training process can be found in Section 5.6.

One surprise while building this system was the consistency with which *question demoting* improved scores for the BOW similarity measures. With this relatively minor change the average correlation between the BOW methods' similarity scores and the student grades improved by up to 0.046 with an average improvement of 0.019 across all eleven semantic features. Table 5.1 shows the results of applying question demoting to the semantic features. When comparing scores using RMSE, the difference is less consistent, yielding an average improvement of 0.002. However, for one measure (tf*idf), the improvement is 0.063 which brings its RMSE score close to the lowest of all BOW metrics.

TABLE 5.1. BOW features with question demoting (QD). Pearson's correlation, root mean square error (RMSE), and median RMSE for all individual questions. Note that the discrepancy on several scores (Lesk, HSO, RES, LCH) is due to a normalization error in an early version of the system.

	MM2009	MM2011					
	r	r	w/ QD	RMSE	w/ QD	Med. RMSE	w/ QD
Lesk	0.363	0.450	0.462	1.034	1.050	0.930	0.919
JCN	0.450	0.443	0.461	1.022	1.026	0.954	0.923
HSO	0.196	0.441	0.456	1.036	1.034	0.966	0.935
PATH	0.441	0.436	0.457	1.029	1.030	0.940	0.918
RES	0.252	0.409	0.431	1.045	1.035	0.996	0.941
Lin	0.392	0.382	0.407	1.069	1.056	0.981	0.949
LCH	0.223	0.367	0.387	1.068	1.069	0.986	0.958
WUP	0.337	0.325	0.343	1.090	1.086	1.027	0.977
ESA Wikipedia	0.468	0.395	0.401	1.031	1.086	0.990	0.955
LSA Wiki-CS	0.463	0.328	0.335	1.065	1.061	0.951	1.000
tf*idf	0.365	0.281	0.327	1.085	1.022	0.991	0.918
Avg.grade	N/A	N/A	N/A	1.097	1.097	0.973	0.973

Looking at the actual distribution of the tf*idf scores, it has been observed that this measure (after isotonic regression) produces scores entirely within the range [2.5 to 5] with

[1]Specifically, the Pool Adjacent Violators Algorithm (PAVA) as implemented by Dr. Razvan Bunescu is used.

over 900 scores between 4.25 and 4.75. Given the bias inherent in this dataset, it is likely to be a good strategy for reducing error, but is not necessarily an indicator of true discriminative ability. For reference, I include here (in Table 5.2) the results of assigning the same grade to every student for all 11 grades awarded by the system (0, 0.5, ..., 5). The average grade (as determined on the training data) is also included for each question. The average grade was selected as a baseline as it is the constant value with the minimum RMSE for the training data.

TABLE 5.2. The simplest baselines possible – always guessing the same value. Note that in this case correlation statistics are undefined, so they are not reported here.

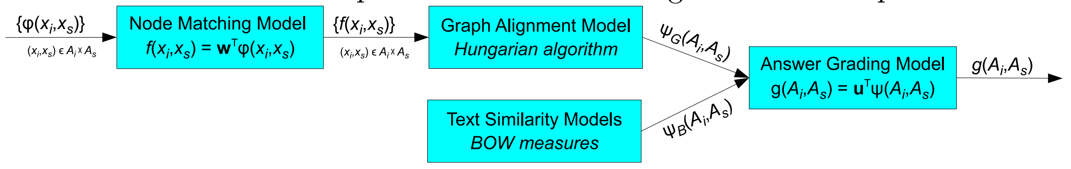
	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	average
RMSE	4.32	3.84	3.37	2.90	2.44	2.01	1.61	1.29	1.11	1.14	1.37	1.10
Accuracy (w/in 0.5)	1.1%	1.6%	2.5%	6.4%	11%	16%	20%	24%	31%	72%	63%	31%

5.2. Alignment Pipeline

In the first stage (Section 5.3), the system is provided with the dependency graphs for each pair of instructor (A_i) and student (A_s) answers. For each node in the instructor's dependency graph, I compute a similarity score for each node in the student's dependency graph. This score is based upon a set of lexical, semantic, and syntactic features applied to both the pair of nodes themselves and their corresponding subgraphs – for instance, the sets of nodes and edges reachable from a starting node assuming that a governor-dependent relationship represents a directional edge from the governor to the dependent. The scoring function is trained on a small set of manually aligned graphs using the averaged perceptron algorithm.

In the second stage (Section 5.4), the node similarity scores calculated in the previous stage are used to weight the edges of a bipartite graph where the nodes of A_i are on one side and the nodes of A_s are on the other. I then apply the Hungarian algorithm to find both an optimal matching and the score associated with such a matching. Question demoting (see Section 5.1) is optionally applied to this step as well (with some modification as described below).

FIGURE 5.1. Pipeline model for scoring short-answer pairs



In the final stage (Section 5.5), SVM-based machine learning is employed to produce an overall grade based upon the alignment scores found in the previous stage as well as the results of several semantic BOW similarity measures (Section 4.1).

5.3. Node Alignments

Dependency graphs for both the student and instructor answers are generated using the Stanford Dependency Parser [27] in collapse/propagate mode. The graphs are further post-processed to propagate dependencies across the "APPOS" (appositive) relation, to explicitly encode negation, part-of-speech, and sentence ID within each node, and to add an overarching ROOT node governing the main verb or predicate of each sentence of an answer. The final representation is a list of (relation, governor, dependent) triples, where governor and dependent are both tokens uniquely described by the tuple (sentenceID:token:POS:wordPosition). For example: (nsubj, 1:provide:VBZ:4, 1:program:NN:3) indicates that the noun "program" is a subject in sentence 1 whose associated verb is "provide."

If the dependency graphs output by the Stanford parser are considered to be directed (minimally cyclic) graphs,[2] it is possible to define for each node x a set of nodes N_x that are reachable from x using a subset of the relations (i.e. edge types)[3]. The term "reachable" is variously defined in four ways to create four subgraphs defined for each node. These are as follows:

- N_x^0 : All edge types may be followed

[2]The standard output of the Stanford Parser produces rooted trees. However, the process of collapsing and propagating dependences violates the tree structure which results in a tree with a few cross-links between distinct branches.

[3]For more information on the relations used in this experiment, consult the Stanford Typed Dependencies Manual at http://nlp.stanford.edu/software/dependencies_manual.pdf

- N_x^1 : All edge types except for subject types, ADVCL, PURPCL, APPOS, PARATAXIS, ABBREV, TMOD, and CONJ
- N_x^2 : All edge types except for those in N_x^1 plus object/complement types, PREP, and RCMOD
- N_x^3 : No edge types may be followed (This set is the single starting node x)

Subgraph similarity (as opposed to simple node similarity) is a means to escape the rigidity involved in aligning parse trees while making use of as much of the sentence structure as possible. Humans intuitively make use of modifiers, predicates, and subordinate clauses when determining that two sentence entities are similar. For instance, the entity-describing phrase "men who put out fires" matches well with "firemen," but the words "men" and "firemen" have less of an inherent similarity. It remains to be determined how much of a node's subgraph will positively enrich its semantics. In addition to the complete N_x^0 subgraph, N_x^1 and N_x^2 were included to tighten the scope of the subtree by first removing more abstract relations, then sightly more concrete relations.

A total of 68 features have been employed to train the machine learning system to compute node-node (or more specifically, subgraph-subgraph) matches. Of these, 36 are based upon the semantic similarity of the four subgraphs defined by $N_x^{[0..3]}$. All eight WordNet-based similarity measures listed in Section 4.1 plus the LSA model[4] are used to produce these features. The remaining 32 features are lexico-syntactic features[5] defined only for N_x^3 and are described in more detail in Table 5.4.

I have used $\phi(x_i, x_s)$ to denote the feature vector associated with a pair of nodes $\langle x_i, x_s \rangle$, where x_i is a node from the instructor answer A_i and x_s is a node from the student answer A_s. A matching score can then be computed for any pair $\langle x_i, x_s \rangle \in A_i \times A_s$ through a linear scoring function $f(x_i, x_s) = \mathbf{w}^T \phi(x_i, x_s)$. In order to learn the parameter vector $\mathbf{w}$, the averaged version of the perceptron algorithm was used [23, 33].

[4]LSA experiments performed on the MM2011 dataset use only the domain-focused subset of a full Wikipedia dump as described in Section 4.2.1. Note also that a disk crash required me to use a newer version of Wikipedia that was 1.8GB after filtering for domain.

[5]Note that synonyms include negated antonyms (and vice versa). Hypernymy and hyponymy are restricted to at most two steps).

TABLE 5.3. Perceptron training for node matching

```
0. set $\mathbf{w} \leftarrow 0$, $\overline{\mathbf{w}} \leftarrow 0$, $n \leftarrow 0$
1. repeat for $T$ epochs:
2.     foreach $\langle A_i; A_s \rangle$:
3.         foreach $\langle x_i, x_s \rangle \in A_i \times A_s$:
4.             if $sgn(\mathbf{w}^T \phi(x_i, x_s)) \neq sgn(A(x_i, x_s))$:
5.                 set $\mathbf{w} \leftarrow \mathbf{w} + A(x_i, x_s)\phi(x_i, x_s)$
6.             set $\overline{\mathbf{w}} \leftarrow \overline{\mathbf{w}} + \mathbf{w}$, $n \leftarrow n + 1$
7. return $\overline{\mathbf{w}}/n$.
```

TABLE 5.4. Subtree matching features used to train the perceptron

Name	Type	# features	Description
RootMatch	binary	5	Is a ROOT node matched to: ROOT, N, V, JJ, or Other
Lexical	binary	3	Exact match, Stemmed match, close Levenshtein match
POSMatch	binary	2	Exact POS match, Coarse POS match
POSPairs	binary	8	Specific X-Y POS matches found
Ontological	binary	4	WordNet relationships: synonymy, antonymy, hypernymy, hyponymy
RoleBased	binary	3	Has as a child - subject, object, verb
VerbsSubject	binary	3	Both are verbs and neither, one, or both have a subject child
VerbsObject	binary	3	Both are verbs and neither, one, or both have an object child
Semantic	real	36	Nine semantic measures across four subgraphs each
Bias	constant	1	A value of 1 for all vectors
Total		68	

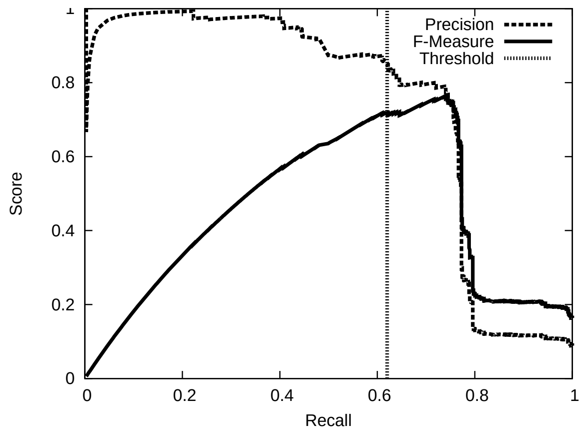
As training data, a subset of the student answers was randomly sampled in such a way that the set was roughly balanced between good scores, mediocre scores, and poor scores. Each node pair $\langle x_i, x_s \rangle$ was then manually annotated as matching, i.e. $A(x_i, x_s) = +1$, or not matching, i.e. $A(x_i, x_s) = -1$. Overall, 32 student answers in response to 21 questions with a total of 7303 node pairs (656 matches, 6647 non-matches) were manually annotated. The pseudocode for the learning algorithm is shown in Table 5.3. Once they had been used to train the perceptron, these 32 student answers were removed from the dataset, were not used as training further along in the pipeline, and were not included in the final results. After training for 50 epochs,[6] the matching score $f(x_i, x_s)$ is calculated (and cached) for each node-node pair across all student answers for all assignments.

For the purpose of this experiment, the scores associated with a given node-node matching are converted into a simple yes/no matching decision where positive scores are considered a match and negative scores a non-match. The threshold weight learned from the bias feature strongly influences the point at which real scores change from non-matches

[6]This value was chosen arbitrarily and was not tuned in any way.

to matches, and given the threshold weight learned by the algorithm, an F-measure of 0.72, with precision(P) = 0.85 and recall(R) = 0.62 can be computed. However, as the perceptron is designed to minimize error rate, this may not reflect an optimal objective when seeking to detect matches. By manually varying the threshold, it is possible to find a maximum F-measure of 0.76, with P=0.79 and R=0.74. Figure 5.2 shows the full precision-recall curve with the F-measure overlaid.

Figure 5.2. Precision, recall, and F-measure on node-level match detection



5.4. Graph Alignment

Once a score has been computed for each node-node pair across all student/instructor answer pairs, I attempt to find an optimal alignment for the answer pair by treating the answer pair as a bipartite graph in which each node in the student answer is represented by a node on the left side of the bipartite graph and each node in the instructor answer is represented by a node on the right side. The score associated with each edge is the score computed for each node-node pair in the previous stage. The bipartite graph is then augmented by adding dummy nodes to both sides which are allowed to match any node with a score of zero. An optimal alignment between the two graphs is then computed efficiently using the Hungarian algorithm. Note that this results in an optimal matching, not a mapping, so that an individual node can be associated with at most one node in the other answer.

At this stage I also compute several alignment-based scores by applying various transformations to the input graphs, the node matching function, and the alignment score itself.

The first and simplest transformation involves the normalization of the alignment score. While there are several possible ways to normalize a matching such that longer answers do not unjustly receive higher scores, I have opted to simply divide the total alignment score by the number of nodes in the instructor answer.

The second transformation scales the node matching score by multiplying it with the *idf*[7] of the instructor answer node, i.e. replace $f(x_i, x_s)$ with $idf(x_i) * f(x_i, x_s)$.

The third transformation – *question demoting* – involves removing from the bipartite graphs, of both the instructor answer and the student answer, any words found in the question. The justification for this was described more in Section 5.1.

The application of these three transformations leads to a total of eight transform combinations, and therefore eight different alignment scores. For a given answer pair (A_i, A_s), the eight graph alignment scores are assembled into a feature vector $\psi_G(A_i, A_s)$.

Before applying any machine learning techniques, I first test the quality of the eight graph alignment features $\psi_G(A_i, A_s)$ independently. Results indicate that the basic alignment score performs comparably to most BOW approaches. The introduction of *idf* weighting seems to degrade performance somewhat, while introducing question demoting causes the correlation with the grader to increase while also increasing RMSE somewhat. The four normalized components of $\psi_G(A_i, A_s)$ are reported in Table 5.5.

TABLE 5.5. Alignment feature/grade correlations using Pearson's r and two RMSE measures. Results are also reported when inverse document frequency weighting (IDF) and question demoting (QD) are used – alone and in conjunction.

	Standard	w/ IDF	w/ QD	w/ QD+IDF
Pearson's r	0.411	0.277	0.428	0.291
RMSE	1.018	1.078	1.046	1.076
Median RMSE	0.910	0.970	0.919	0.992

[7]Inverse document frequency, as computed from the British National Corpus (BNC)

5.5. Machine Learning

In the hopes of exploiting the various advantages of both bag-of-words approaches (Section 4.1) and the structure-aware alignment module (Section 5.4), each of these are used as features in a Support Vector Machine (SVM) to produce a combined real-number grade. In addition, an Isotonic Regression (IR) model is built to transform the computed output scores onto the original [0..5] scale for ease of comparison.

An SVM, in its simplest form, is a maximum margin binary classifier. It takes a series of inputs (in one of two categories) and maps each one to a point in high dimensional space. It then finds the hyperplane in that space that separates data points based upon their category. If the points are linearly separable, the hyperplane is guaranteed to be as far from a point in either dataset as possible. Unseen inputs can then be classified based upon which side of this hyperplane they fall on. In this work, I utilize and compare three extensions to the SVM model – SVM for regression (SVR), ranking SVM, and support vector ordinal regression. An in-depth analysis of these extensions is beyond the scope of this work, but interested readers are here referred to the relevant literature [21, 49, 92].

5.5.1. SVM Features and Implementations

The alignment scores $\psi_G(A_i, A_s)$ are combined with the scores $\psi_B(A_i, A_s)$ from the lexical semantic similarity measures into a single feature vector $\psi(A_i, A_s) = [\psi_G(A_i, A_s)|\psi_B(A_i, A_s)]$. The feature vector $\psi_G(A_i, A_s)$ contains the eight alignment scores found by applying the three transformations in the graph alignment stage. The feature vector $\psi_B(A_i, A_s)$ consists of eleven semantic features – the eight knowledge-based features plus LSA, ESA and a vector consisting only of tf*idf weights – both with and without question demoting. Thus, the entire feature vector $\psi(A_i, A_s)$ contains a total of 30 features.

An input pair (A_i, A_s) is then associated with a grade $g(A_i, A_s) = \mathbf{u}^T\psi(A_i, A_s)$ computed as a linear combination of features. The weight vector $\mathbf{u}$ is trained to optimize performance in three scenarios:

Regression: An SVM model for regression (SVR) is trained using as target function the grades assigned by the instructors using the libSVM [8] implementation of SVR, with tuned parameters.

Ranking: An SVM model for ranking (SVMRank) is trained using as ranking pairs all pairs of student answers (A_s, A_t) such that $grade(A_i, A_s) > grade(A_i, A_t)$, where A_i is the corresponding instructor answer using the SVMLight [9] implementation of SVMRank with tuned parameters.

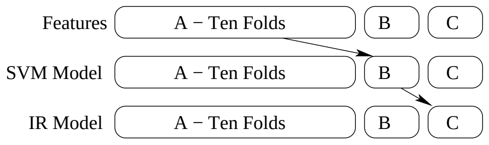
Ordinal Regression: An SVM model for ordinal regression (SVORIM) is trained using as a target function the grades mapped to a [1..11] scale. I employ the package implemented by Hsuan-Tien Lin[10] with tuned parameters.

In all cases, the parameters (for cost C and tube width ϵ) were found using a grid search. At each grid point, the training data was partitioned into 5 folds which were used to train a temporary SVM model with the given parameters. The regression packages selected the grid point with the minimal mean square error (MSE), while the SVMRank package tried to minimize the number of discordant pairs. The parameters found were then used to score the test set – a set not used in the grid training.

5.6. Results

The SVM components of the system are run on the full dataset, retraining once for each of the 10 assignments and 2 examinations (for a total of 12 assignments). Each assignment is scored independently with ten of the remaining eleven assignments used to train the SVM system. For each assignment, one additional assignment is held out for later use in the development of an isotonic regression model (see Figure 5.3).

FIGURE 5.3. Dependencies of the SVM/IR training stages



[8]http://www.csie.ntu.edu.tw/˜cjlin/libsvm/

[9]http://svmlight.joachims.org/

[10]http://www.work.caltech.edu/ htlin/program/libsvm/

All SVM models were trained using a linear kernel.[11] Results from all three SVM implementations are reported in Table 5.6 along with a selection of other measures. Note that the RMSE score was computed after performing isotonic regression on the SVMRank results, but that it was unnecessary to perform an isotonic regression on the SVR and SVORIM results as the system was trained to produce a score on the correct scale.

I report the results of running the systems on three subsets of features $\psi(A_i, A_s)$: BOW features $\psi_B(A_i, A_s)$ only, alignment features $\psi_G(A_i, A_s)$ only, or the full feature vector (labeled "Hybrid"). Finally, three subsets of the alignment features are used: only unnormalized features, only normalized features, or the full alignment feature set.

TABLE 5.6. The results of the SVM models trained on the full suite of BOW measures, the alignment scores, and the hybrid model. The terms "normalized," "unnormalized," and "both" indicate which subset of the 8 alignment features were used to train the SVM model. For ease of comparison, each section includes the scores for the inter-annotator agreement (IAA), the "Average grade" baseline, and two of the top performing BOW metrics – both with question demoting.

						Unnormalized		Normalized		Both	
	IAA	Avg. grade	tf*idf	Lesk	BOW	Align	Hybrid	Align	Hybrid	Align	Hybrid
								SVMRank			
Pearson's r	0.586		0.327	0.450	0.480	0.266	0.451	0.447	0.518	0.424	0.493
RMSE	0.659	1.097	1.022	1.050	1.042	1.093	1.038	1.015	0.998	1.029	1.021
Median RMSE	0.605	0.973	0.918	0.919	0.943	0.974	0.903	0.865	0.873	0.904	0.901
Accuracy (w/in 0.5)	0.807	0.309	0.639	0.644	0.634	0.534	0.626	0.644	0.662	0.624	0.646
								SVR			
Pearson's r	0.586		0.327	0.450	0.431	0.167	0.437	0.433	0.459	0.434	0.464
RMSE	0.659	1.097	1.022	1.050	0.999	1.133	0.995	1.001	0.982	1.003	0.978
Median RMSE	0.605	0.973	0.918	0.919	0.910	0.987	0.893	0.894	0.877	0.886	0.862
Accuracy (w/in 0.5)	0.807	0.309	0.639	0.644	0.621	0.493	0.626	0.636	0.632	0.630	0.617
								SVORIM			
Pearson's r	0.586		0.327	0.450	0.454	0.076	0.462	0.447	0.490	0.444	0.502
RMSE	0.659	1.097	1.022	1.050	1.018	1.158	1.012	1.034	0.990	1.035	0.978
Median RMSE	0.605	0.973	0.918	0.919	0.957	1.044	0.962	0.906	0.898	0.915	0.915
Accuracy (w/in 0.5)	0.807	0.309	0.639	0.644	0.641	0.703	0.640	0.658	0.652	0.663	0.666

5.7. Error Analysis

In this section, I explore the limitations of the existing system and seek to detect any misleading artifacts in my experimental setup by performing an in-depth analysis of

[11]The SVR system was also run using quadratic and radial-basis function (RBF) kernels, but the results did not show consistent improvement over the simpler linear kernel.

individual errors. In the interest of brevity, I consider only the system using the libSVM package with all features available for learning (bag-of-words, normalized alignment data, and unnormalized alignment data). This system configuration resulted in the lowest error rate (RMSE), is transparent given the weight vector, and utilizes the full set of features available. In theory, this system should be making the best decisions.

5.7.1. Analysis Methodology

When using this system, 567 student responses (24.9%) were given a grade more than 1 point away from the average score of the two graders. Of these, forty-eight (48) were more than 2.5 points away from the correct score. I have analyzed the responses associated with these 48 errors as well as 26 responses chosen randomly with an error between 1 and 1.75, and 26 responses chosen randomly with an error between 1.75 and 2.5. These 100 errors can be attributed to twelve different causes or limitations, which is described (with examples) below. The frequency of each type of error can be found in Table 5.7 lit based upon the magnitude of the error.

TABLE 5.7. Counts for each error type. The pattern of errors differs based upon magnitude of error, so they are shown divided here.

	1.0 to 1.75	1.75 to 2.5	over 2.5	Full Set
Unreachable Lows	1	8	23	32
BOW Tricked	6	14	9	29
No Answer	0	0	15	15
Within Grade Range	5	1	0	6
Symbolic Response	1	3	0	4
Alignment Penalty	3	0	0	3
Unmatched Phrases	3	0	0	3
Over-normalization	3	0	0	3
Contradictions	1	0	1	2
Bounds Irregularity	1	0	0	1
Converse Topic	1	0	0	1
Spelling Error	1	0	0	1
Total	26	26	48	100

5.7.2. Error Descriptions

Two of the error categories involve simple mistakes that can and should be handled in post-processing.

[No Answer] Whenever a student fails to respond to a question, the content-delivery system produces the text "Not Answered." The system treats this like any other text, but should instead detect this as a special case. This is responsible for 15 large errors.

[Bounds Irregularity] In this case, the system has produced a score above 5.0 (or below 0). Only one case of this error was found in this sample, but it certainly contributes to error unnecessarily and may also affect lower-magnitude errors (with less than 1.0 error).

Some errors are due to well-understood natural language processing issues that have not been integrated into the current system.

[Spelling Errors] An incorrectly spelled word failed to match correctly using either the BOW measures or the alignment measures. Example: (7.4.21) "by reference" fails to match "by refrenece" [sic].

[Unmatched Phrases] Multi-word phrases fail to produce an alignment match. Example: (2.5.9)

- Question: How many constructors can be created for a class?
- Answer: Unlimited number.
- Student: as many as you want

The phrase "as many as you want" should have been matched with the word "unlimited" but the structures of the two phrases are so different as to make detecting this match difficult.

Two others are more difficult to approach, in that some knowledge of the outside word is required in order to link an answer with the given response.

[Symbolic Response] In some cases, students would respond using mathematical symbols that the system is not prepared to analyze. Example: (12.10.7)

- Question: How many steps does it take to search a node in a binary search tree?
- Answer: The height of the tree.
- Student: 2n̂ where n is the # of levels the binary tree has

[Converse Topic] The instructor answer and the student answer solve the same question from different sides and have very little semantic overlap. Example: (7.2.25)

- Question: What is the main advantage of linked lists over arrays?
- Answer: The linked lists can be of variable length.
- Student: The size of array is restricted to declaration. Insertion or Deletion of values in middle of // array is not possible.

While the student technically described what the disadvantages were associated with an array, both graders gave a full score to this student, but the system could not reasonably infer the the student and instructor gave the same answers.

One category suggests the difficulty of the grading task itself.

[Within Grade Range] The two graders gave vastly different scores and the system was very near to one of the graders. Example: (4.4.7)

- Question: What is the difference between an array declared as static, and one that is not?
- Answer: The arrays declared as static live throughout the life of the program; that is, they are

  initialized only once, when the function that declares the array it is first called.
- Student: a static array has pre-runtime size and that size cannot be changed. A dynamic array gets its size at runtime.
- Scores: 1 (Grader 1), 5 (Grader 2), 3 (Average), 4.02 (System)

Whenever a wide gap exists between the two annotators (greater than 1.0 almost 20% of the time), it is impossible to know whether the system really should support one grader over the other, or if it should be in between. Sometimes assigning a response to this error was a judgement call, as it appeared that one of the annotators fell into the trap of skimming or looking for lexical overlap. Human error or inconsistency is always a possibility, which is one advantage of an automated system.

Three types of error reiterate the limitations of the BOW-approach and lay bare the need for a deeper, syntactic analysis of the text.

[Contradictions] The student produces a correct answer, but then continues the answer in such a way that the answer becomes invalid. Many approaches detect the correct answer without being aware of the spoilage associated with the contradiction. Example: (10.3.7)

- Question: What is a leaf?
- Answer: A node that has no children.
- Student: A leaf is a node with children, it is a terminating node.

The phrase "terminating node" would be correct on its own, but the fact that the student explicitly negated the instructor answer was decisive.

[BOW Tricked] This is a very broad category, that simply indicates that many of the features (esp. bag-of-words features) believed that student answer was very similar to the instructor answer. However, this is misleading due to a deeper analysis of the text. Examples: (3.3.13) and (8.7.10)

- Question: How does the compiler handle inline functions?
- Answer: It makes a copy of the function code in every place where a function call is made.
- Student: it treats them as the same function.
- Question: What operations would you need to perform to find a given element on a stack?
- Answer: Pop all the elements and store them on another stack until the element is found, then // push back all the elements on the original stack.
- Student: pop and push

In these two cases, the presence of several high-content words "function" or "push"/"pop" tricked the system into believing that the student had provided a correct answer, without realizing that more information was needed.

[Over-normalizations] The system detects a correct and crucial alignment, but the alignment scores are so low as to be practically ignored. Example: (12.5.20)

- Question: What is the advantage of linked lists over arrays?

- Answer: Linked lists are dynamic structures, which allow for a variable number of elements to be // stored.
- Student: linked lists do not have a memory constraint other than total memory
- Alignments: Among other things the phrases "variable" and "do not have a memory constraint" match, but // the benefits of the alignment are minimal due to the many elements that do not match.

Finally, two severe limitations are due to the selection of features available for learning. All of these features should correlate positively with textual similarity – i.e. as the measures become more positive, the chance of a correct score should improve. This means that in the absence of any evidence of similarity the feature values will all be very close to zero, and the final result will be very close to the SVM bias value. For all 12 models built (one per fold), the bias value is between 3.2 and 3.8, so final scores tend to be clustered within this region. Due to the skewed nature of the training data – few examples with grades below 3 – this is a good learning strategy for minimizing error, but does not clearly indicate the ability to distinguish good answers from poor ones.

[Unreachable Lows] Since there is no feature meant to correlate with textual dissimilarity, it is not possible for the SVM system to produce very low scores. Lack of similarity results in scores near the bias.

[Alignment Penalty] Roughly half of the feature weights, including those of many of the simple alignment features are negative. In particular, the weight associated with a non-demoted alignment, unnormalized, with IDF scaling is below -1.0 in all models. In some cases, this results in a good alignment negatively impacting the overall score. Example: (2.4.15)

- Question: When does C++ create a default constructor?
- Answer: If no constructor is provided, the compiler provides one by default. If a constructor is defined // for a class, the compiler does not create a default constructor.
- Student: When no constructor exists when one is needed, a parameterless default

constructor is declared.

- Alignment Scores: 0.40*-0.38 + 0.34*0.69 + 0.28*-1.65 + 0.21*0.27 + 0.21*0.54 + 0.26*0.07 + 0.03*0.66 + 0.02*0.42 = -0.17

CHAPTER 6

SEMEVAL2012 AND OTHER DATASETS

Excluding a few grading-specific modifications (such as pseudo-relevance feedback and question demoting) the techniques described in this thesis are equally applicable to non-grading short text similarity tasks. In this chapter, I explore the application of these techniques to the closely related tasks of detecting semantic textual similarity, textual entailment, and paraphrase. One benefit to such a study is that several datasets are publicly available and have been widely used by researchers for a number of years (see Section 3.2). Many existing systems have been tested using each of these datasets which provides a chance to compare this system's results with those of others in the field.

I first detail a joint submission to the SemEval 2012 Semantic Text Similarity (STS) Task, which included many of the components described in Chapters 4 and 5. I then explore the capabilities of our joint system disentangled from the contributions of the remainder of the team. Finally, I show the results of applying this system to the rest of the datasets described in Section 3.2 and analyze the results.

6.1. SemEval 2012 - Semantic Textual Similarity (Task 6)

The STS task website[1] describes the semantic textual similarity (STS) task as being related to both textual entailment and paraphrase detection, but different in two key ways. First, STS assumes a bidirectional relationship, which is not the case for entailment. Given the two sentences "Booth killed Lincoln" and "Lincoln is dead," it is clear that the first entails the second, but not the other way around. Second, STS assumes a graded response where both entailment and paraphrase are binary decisions. For the SemEval task, the gold standard scores are within the range [0..5].

Each team was allowed to train using the development set for one month before the test data was released. Teams were allowed only five days with the test data before they

[1]http://www.cs.york.ac.uk/semeval-2012/task6/

were required to submit their predicted scores for the data set. Up to three sets of scores per team were allowed.

Thirty-five teams submitted a total of 88 system runs.[2] At the time of submission, the method of evaluation was not entirely clear, but after the fact, the contest organizers indicated that evaluation, using Pearson's correlation coefficient, would be carried out in three ways. First, all five components of the dataset were concatenated and correlation across the whole dataset was found (labeled "ALL" in Table 6.1). Second, a correlation coefficient was calculated for each of the five components individually before finding the weighted mean (reported as "Mean") in Table 6.1). Finally, the output for each dataset was separately normalized, using the linear least squares method before computing the Pearson's correlation coefficient for the concatenated dataset. Due to its complexity, and for reasons of space, I have elected not to include this score in Table 6.1.

6.1.1. Our STS Submission

Our team consisted of three individuals – Carmen Banea, Dr. Samer Hassan, and myself – under the supervision of Dr. Rada Mihalcea. Our submission [12] can be described as a combination of three subsystems which independently produced a set of features used as training for a machine learning system. These three feature components were the knowledge-based similarity scores described in Section 4.1.1, the graph alignment scores described in Section 5.4, and a set of corpus-based similarity features including tf*idf, latent semantic analysis (LSA), explicit semantic analysis (ESA), and salient semantic analysis (SSA), which was the primary contribution of my teammates. An analysis of SSA is beyond the scope of this work, but details can be found in our system description paper [12] and, to a greater extent, in the publications of my colleague Dr. Hassan [41].

It should be pointed out here that for both the knowledge-based and graph-alignment features the system setup has changed slightly from previous chapters. The Hirst & St. Onge metric was discarded for reasons of time, both as a simple feature and as a feature in the alignment system. Our LSA models were retrained on a newer Wikipedia dump (again set to

[2]Some of these were not published officially due to being submitted late or other issues.

the CS domain as described in Section 4.2.1) and are now built using the SemanticVectors implementation (http://code.google.com/p/semanticvectors/) instead of Infomap. These changes are maintained for all experiments reported in this chapter.

Given this set of real-valued features (knowledge-based, corpus-based, and graph-alignment), the training of our machine learning systems proceeded in one of two ways [3]. Either the full training set was used to train our machine learning models (labeled as "Combined" training) or the individual components of the dataset were used for the corresponding component in the testing set (labeled as "Individual" training). For instance, examples taken from the MSRpar corpus in the testing dataset were scored using a model built only using examples from the Microsoft Research Paraphrase (MSRP) corpus in the training dataset. The same is true for the MSRvid and SMTeuroparl components. For the two surprise datasets (OnWN and SMTnews), the combined training dataset was used in both cases.

In addition, two different machine learning algorithms were employed in our submission. First, we used support vector regression (SVR) with a Pearson VII function-based kernel. We also attempted to produce a model using the M5P decision tree algorithm. Since each team was permitted to submit up to three runs, we submitted the results of the SVR system using two methods of training (IndividualRegression and CombinedRegression) as well as the M5P system using only the "Individual" method of training (IndividualDecTree). Results for these submissions (as supplied by the contest organizers) can be found at the top of Table 6.1.

Our top submission (IndividualRegression) was ranked 5th among all 88 submissions according the the "ALL" evaluation criterion and our top correlation coefficient (0.7846) required around 5% improvement to surpass the challenge's top ranked submission (0.8239)[4].

For the sake of comparison, Table 6.1 includes also the results of running all of the similarity metrics described in Chapters 4 and 5 on the testing dataset using the "Individual" training method where applicable. For the machine learning component reported here, I have

[3]Note that this is different from the machine learning described in Section 5.5 and was carried out by my colleague, Carmen Banea.

[4]Full results can be found at: http://www.cs.york.ac.uk/semeval-2012/task6/index.php?id=results-update

TABLE 6.1. Detailed evaluation results for the SemEval 2012 Semantic Text Similarity task. The top section matches the scores published by the task organizers. The remaining sections indicate the scores (and ranks) that would have been achieved if the indicated system were submitted to this task.

Run	ALL	Rank	Mean	RankMean	MSRpar	MSRvid	SMTeuroparl	OnWN	SMTnews
$Team - IndividualRegression$	0.7846	5	0.6162	13	0.5353	0.8750	0.4203	0.6715	0.4033
$Team - IndividualDecTree$	0.7677	9	0.5947	25	0.5693	0.8688	0.4203	0.6491	0.2256
$Team - CombinedRegression$	0.7418	14	0.6159	14	0.5032	0.8695	0.4797	0.6715	0.4033
Wordnet JCN	0.4882	68	0.5323	55	0.4825	0.6494	0.4977	0.6216	0.2773
Wordnet LCH	0.3287	85	0.5016	63	0.5103	0.5625	0.4952	0.5445	0.2975
Wordnet Lesk	0.4775	69	0.5507	51	0.4742	0.6401	0.5142	0.6404	0.3998
Wordnet Lin	0.4118	81	0.5021	63	0.4864	0.5698	0.4987	0.5740	0.2730
Wordnet Path	0.4752	69	0.5491	52	0.4988	0.6652	0.5013	0.6206	0.3463
Wordnet Res	0.4200	78	0.5281	56	0.4777	0.6083	0.5069	0.6175	0.3284
Wordnet WUP	0.2312	89	0.4454	69	0.4939	0.4414	0.4716	0.4848	0.2572
LSA	0.5657	53	0.5380	53	0.4289	0.6821	0.3965	0.6513	0.4218
ESA	0.5600	55	0.4760	67	0.2319	0.7362	0.3634	0.5771	0.3855
tf*idf	0.5014	63	0.4752	67	0.3646	0.6880	0.4751	0.4579	0.3158
$Alignment$	0.2216	89	0.3899	79	0.3532	0.4754	0.2220	0.4500	0.3780
$Alignment - Norm$	0.3133	86	0.4769	67	0.4469	0.4560	0.4395	0.5288	0.5179
$Alignment - IDF$	0.1450	89	0.3253	86	0.0896	0.6787	0.1676	0.3265	0.2827
$Alignment - IDF + Norm$	0.2554	89	0.2887	87	0.0453	0.6152	0.2672	0.2462	0.2373
$SVRBOW - only$	0.7605	10	0.6008	22	0.5409	0.8051	0.5078	0.6250	0.3910
$SVRUnnormalizedAlign$	0.6510	37	0.4149	74	0.3586	0.7013	0.0018	0.4519	0.3876
$SVRNormalizedAlign$	0.7103	20	0.5209	58	0.4492	0.6453	0.4307	0.5179	0.5312
$SVRFullAlign$	0.6722	23	0.5293	55	0.4416	0.7120	0.4372	0.5021	0.5074
$SVRUnnormalizedAlign + BOW$	0.5558	56	0.5192	59	0.1467	0.8311	0.5156	0.6363	0.4168
$SVRNormalizedAlign + BOW$	0.7604	10	0.5912	29	0.4877	0.8276	0.4880	0.6246	0.3976
$SVRFullAlign + BOW$	0.6409	36	0.5292	55	0.2077	0.8323	0.4891	0.6352	0.4106

considered only the support vector regression model (using the libSVM package). I have used the parameters found after performing a grid search on the MM2011 dataset as described in Section 5.5.1 and did not refine these parameters for any of the experiments reported in this chapter[5].

As expected, the runs our team submitted performed best on the whole, but were outperformed (or matched) in some cases at the finer-grained level. Altogether, the regression models described in Section 5.5 came closest to the team submission (esp. $SVRBOW-only$ and $SVRNormalizedAlign+BOW$). The only difference between these systems and the one submitted was the lack of SSA as an available feature, a different support vector regression learning implementation (libSVM vs Weka), and a different implementation of LSA/ESA along with different training corpora. As it stands, my best SVR implementation would have achieved 10th among all submissions according to the "ALL" evaluation metric.

[5]The cost and epsilon values were both set to 0.5.

For two of the datasets (SMTeuroparl and SMTnews), my system's metrics significantly outperformed the team submissions. For SMTeuroparl, almost all of the knowledge-based measures (including tf*idf) outperformed the team's machine learning systems and one of the alignment features ($Alignment - Norm$) performed just as well. For the SMTnews corpus, alignment seemed to play an even more important part. The same top-performing alignment feature ($Alignment - Norm$) here outperformed the best team submission (by 28%), the best knowledge-based measure (by 30%), and the best corpus-based measure (by 23%). Of course, this measure was itself improved upon by using machine learning, though adding BOW features degraded performance here as well. Had my best measure ($SVRNormalizedAlign$) been submitted it would have ranked 4th on the SMTnews corpus.

One may theorize that alignment has an especially important role to play in the evaluation of machine translation (disproportionate to its role in textual similarity). For a translation to be correct, it is much more important for the sentence structure to be maintained than is the case for other similarity tasks (e.g. paraphrase). It may also be the case that, in the context of machine translation, false synonyms due to incorrect word-sense disambiguation (which can confound purely knowledge-based similarity models) are particularly problematic, and so reducing the influence of the word-similarity models may yield an advantage.

6.2. Additional Experiments

One of the long-term goals of this project has been to expand beyond computer-aided assessment (which has a very narrow field of utility) into more general natural language processing (NLP) tasks thus allowing lessons learned here to be applied more broadly. With that in mind, I have instigated a portability study, in addition to the SemEval task, that will provide a way to gage the effectiveness of my methodology when applied to the tasks of detecting paraphrase, textual entailment, and textual similarity in general. All systems that are listed for comparison in the following sections are described in Section 2.3. The system setup for these experiments is the same as that described in the previous section.

TABLE 6.2. Li30 - Pairwise similarity of Miller-Goodenough terms

System	Pearson's r	RSME
Wordnet JCN	0.8518	0.8114
Wordnet LCH	0.8293	0.7048
Wordnet Lesk	0.8671	0.7839
Wordnet Lin	0.9056	0.5367
Wordnet Path	0.8647	0.7111
Wordnet Res	0.8848	0.6728
Wordnet WUP	0.8044	0.7256
LSA	0.7637	0.8941
ESA	0.7522	0.9523
tf*idf	0.7517	0.7845
$Alignment$	0.6858	0.9411
$Alignment - Norm$	0.7141	0.9108
$Alignment - IDF$	0.6425	0.9604
$Alignment - IDF + Norm$	0.5777	1.0187
$SVRBOW - only$	0.8794	0.5188
$SVRUnnormalizedAlign$	0.5042	0.9422
$SVRNormalizedAlign$	0.6383	0.8409
$SVRFullAlign$	0.7528	0.7172
$SVRUnnormalizedAlign + BOW$	0.8855	0.5062
$SVRNormalizedAlign + BOW$	0.8745	0.5309
$SVRFullAlign + BOW$	0.8777	0.5236
SSA [41]	0.881	-
Roget's 1987 Thesaurus [52]	0.8725	-
OMIOTIS [98]	0.856	-
STS [47]	0.853	-
Dynamic Time Warping [63]	0.841	-
Term Pair Heuristic [13]	0.83	-
STASIS [60]	0.816	-

6.2.1. Experiments on the Li30 Similarity Dataset

Beyond SemEval, the first dataset under consideration was introduced by Li et al. [60] and despite its small size has been widely adopted in the text similarity community. A description of the dataset can be found in Section 3.2. For experiments on this dataset scored on a [0..4] scale, I report Pearson's correlation coefficient (as do most other researchers to use this dataset) as well as the root mean squared error (RMSE). In order to compute RMSE, all simple metrics (not including any SVR systems) are placed onto the appropriate scale using isotonic regression which was performed by taking 5 folds and using 4 folds as training to label the remaining 1 fold. SVR was trained using 5 folds in the same manner. Results can be found in Table 6.2.

It would appear that the SVR system performs comparably to state-of-the-art systems that have been evaluated on this dataset. For the most part, it can be said that the machine-learning systems outperformed the simple measures except that the Lin measure seems to comfortably surpass even the top-performing systems. It is not clear why this should be the case, and the small size of the dataset suggests that this score may simply be an outlier.

In general the knowledge-based measures outperformed their corpus-based counterparts as well as the alignment features. Very little improvement is observed when combining the alignment features with the BOW features in the SVR system. However, an analysis performed on this dataset [13] suggests that sentence similarity (which is sensitive to changes in structure) is not really what the annotators were measuring. Rather, the term similarities (recall that the sentences here are definitions of the terms in the Rubenstein and Goodenough corpus) seem to have been germane to the task during annotation. It is thus not surprising that a consideration of sentence structure may be less advantageous here than on a less idiosyncratic dataset.

6.2.2. Experiments on the Lee50 Short Document Similarity Dataset

Arguably turning away from the sentence-similarity task, the system's performance was next assessed on the Lee50 dataset. A description of this dataset can be found in Section 3.2. For experiments on this dataset scored on a [0..1] scale, I report Pearson's correlation coefficient (as do most other researchers to use this dataset) as well as the root mean squared error (RMSE). Training for isotonic regression or SVR were performed as described in Section 6.2.1. Results can be found in Table 6.3.

From these results, one can see that the SVR systems far surpass the simpler metrics and comfortably outperform the state-of-the-art systems that have evaluated on this dataset. It can also be said that while alignment only scores (even with machine learning) can not be considered high quality, their addition to the BOW-only SVR model yields a consistent if subtle improvement for both evaluation metrics.

Also of note is the performance of the vector-based measures, especially ESA and tf*idf. It may be concluded that as the size of the texts increases, the discriminative abilities

TABLE 6.3. Lee50 - "short" document similarity corpus

System	Pearson's r	RSME
Wordnet JCN	0.7334	0.6407
Wordnet LCH	0.7178	0.6534
Wordnet Lesk	0.7331	0.6291
Wordnet Lin	0.7181	0.6418
Wordnet Path	0.7401	0.6281
Wordnet Res	0.7284	0.6368
Wordnet WUP	0.6611	0.6655
LSA	0.6611	0.6641
ESA	0.7301	0.6003
tf*idf	0.7506	0.5638
$Alignment$	0.4702	0.7350
$Alignment - Norm$	0.4964	0.7375
$Alignment - IDF$	0.5835	0.6755
$Alignment - IDF + Norm$	0.5980	0.6783
$SVRBOW - only$	0.7856	0.6110
$SVRUnnormalizedAlign$	0.5784	0.8247
$SVRNormalizedAlign$	0.5914	0.8075
$SVRFullAlign$	0.5951	0.8028
$SVRUnnormalizedAlign + BOW$	0.7929	0.6016
$SVRNormalizedAlign + BOW$	0.7923	0.6030
$SVRFullAlign + BOW$	0.7930	0.6013
WikiWalk [106]	0.766	-
ESA-Gabrilovich [35]	0.72	-
SSA [41]	0.684	-
LSA-Lee et al. [56]	0.6	-

of sparse metrics such as tf*idf become more useful. On the other hand, I note that tf*idf has, in addition to a solid correlation score, the absolute best error rate of all metrics considered (by a wide margin). This suggests that rather than exhibiting a true advantage, the tf*idf measure is exploiting the dataset's skew towards low similarity scores.

6.2.3. Experiments on the RTE-3 Textual Entailment Dataset

I next sought to apply the techniques described in this thesis (many of which were inspired by work in RTE) to the currently well-studied task of entailment detection using the RTE-3 dataset which many quality systems have used to report their results. A description of this dataset can be found in Section 3.2. For experiments on this dataset scored with either a "yes" or a "no," I follow existing work in reporting accuracy, precision (in detecting entailment), recall, and the F1-measure. Rather than using an support vector machine (SVM) regression model which would produce a real-valued score, I instead use the SVM

TABLE 6.4. RTE-3 - Textual entailment evaluated on the test portion of the corpus (with SVM classification)

System	Accuracy	Precision	Recall	F-measure
Baseline - guess "yes"	0.5000	0.5000	1.0000	0.6667
Wordnet JCN	0.6238	0.6004	0.7971	0.6849
Wordnet LCH	0.6350	0.6113	0.7922	0.6901
Wordnet Lesk	0.6350	0.6097	0.8020	0.6923
Wordnet Lin	0.6225	0.5909	0.8582	0.6999
Wordnet Path	0.6363	0.6155	0.7751	0.6861
Wordnet Res	0.6263	0.6049	0.7824	0.6823
Wordnet WUP	0.6350	0.6122	0.7873	0.6888
LSA	0.6100	0.6047	0.6919	0.6454
ESA	0.5938	0.5959	0.6455	0.6197
tf*idf	0.6225	0.6071	0.7482	0.6703
$Alignment$	0.5975	0.5973	0.6601	0.6272
$Alignment - Norm$	0.5800	0.5629	0.8093	0.6640
$Alignment - IDF$	0.5763	0.5589	0.8234	0.6660
$Alignment - IDF + Norm$	0.5900	0.6085	0.5623	0.5845
$SVCBOW - only$	0.6325	0.6152	0.7506	0.6762
$SVCUnnormalizedAlign$	0.5788	0.6154	0.4694	0.5326
$SVCNormalizedAlign$	0.5713	0.5982	0.4988	0.5440
$SVCFullAlign$	0.5713	0.6025	0.4743	0.5308
$SVCUnnormalizedAlign + BOW$	0.6363	0.6185	0.7531	0.6792
$SVCNormalizedAlign + BOW$	0.6350	0.6149	0.7653	0.6819
$SVCFullAlign + BOW$	0.6325	0.6162	0.7457	0.6748
8 Hickl et al. [43]	0.8038	0.8815	-	-
COGEX [97]	0.7225	0.6741	0.8878	0.7663
Nielsen et al. [75]	0.671	-	-	-
Tree Skeletons [102]	0.669	-	-	-
NatLog [20, 66]	0.6362	0.6374	0.6732	0.6548

classification (SVC) model (in the libSVM package) to produce scores in [0,1]. Training for the SVM was performed by using the training component of the dataset to build a model and using the testing component for evaluation. Results are reported in Table 6.4.

Unlike the other datasets, the results reported here are far below the best rated system reported at the RTE-3 Challenge itself [43], though it is competitive with several of the less exceptional systems from the challenge including the Natural Logic system produced at Stanford [66]. To some extent, this is surprising since entailment decisions are much more dependant upon structural issues than paraphrase or general similarity. However, this dataset may also include several "gotcha" type issues where the ability to detect contradiction (even in the presence of similarity) may be required to make an intelligent entailment decision.

That said, the SVC systems that included the alignment features slightly outperformed the BOW-only SVC model in precision, recall, and F-measure, though this does not appear to be a significant improvement. Also worth mentioning is the quality of a generally poor measure ($SVCUnnormalizedAlign$). In this case, the measure produced scores with a very high precision, suggesting that it was able to rate very structurally similar scores better than the other measures under consideration.

6.2.4. Experiments on the MSRP Paraphrase Dataset

Finally, I consider the Microsoft Research Paraphrase Corpus (MSRP) which is easily the most widely used of the datasets employed here. A description of this dataset can be found in Section 3.2. For experiments on this dataset scored with either a "yes" or a "no," I follow existing work in reporting accuracy, precision (in detecting paraphrase), recall, and the F1-measure. Training for the SVM classifier was performed as described in Section 6.2.3. Results are shown in Table 6.5.

Before discussing the results of the experiments on this dataset, I here refer to a study performed on it [13] which points out that all the best reported systems have approximately the same F1-measure (0.80-0.81) despite surface differences in accuracy, precision, or recall. The authors of this study suggest that this indicates an upper limit to the effectiveness of similarity-based systems as applied to paraphrase. Only systems that were specifically designed for paraphrase such as Finch et al. [32] have been able to break this barrier reporting an F-measure around 0.83.

With this in mind, my best SVC systems (as well as the Lin measure and uncharacteristically, the $Alignment - Norm$ measure) achieve an F-measure in the 0.81-0.82 range. The SVC systems that made use of alignment features are characterized by a high recall here compared to the knowledge-based measures which display high precision. Although it is outside the scope of this work, it would be interesting to perform an error analysis on this dataset, to see what types of sentence pairs are enforcing this barrier (at most 0.75 accuracy) on so many systems when the most-common-case baseline is relatively high at 0.66.

TABLE 6.5. MSRP - Evaluated on the test portion of the Microsoft paraphrase corpus (with SVM classification)

System	Accuracy	Precision	Recall	F-measure
Baseline - guess "yes"	0.6649	0.6649	1.0000	0.7987
Wordnet JCN	0.7154	0.7500	0.8586	0.8007
Wordnet LCH	0.7136	0.7314	0.9005	0.8072
Wordnet Lesk	0.7107	0.7407	0.8700	0.8002
Wordnet Lin	0.7316	0.7493	0.8970	0.8165
Wordnet Path	0.7171	0.7435	0.8778	0.8051
Wordnet Res	0.7206	0.7476	0.8761	0.8067
Wordnet WUP	0.7125	0.7420	0.8709	0.8013
LSA	0.6968	0.7125	0.9127	0.8003
ESA	0.6725	0.6785	0.9651	0.7968
tf*idf	0.6800	0.6785	0.9869	0.8041
$Alignment$	0.7032	0.7188	0.9101	0.8032
$Alignment - Norm$	0.7177	0.7323	0.9075	0.8106
$Alignment - IDF$	0.6649	0.6653	0.9991	0.7987
$Alignment - IDF + Norm$	0.6655	0.6665	0.9956	0.7985
$SVCBOW - only$	0.7206	0.7271	0.9276	0.8152
$SVCUnnormalizedAlign$	0.6649	0.6647	1.0000	0.7986
$SVCNormalizedAlign$	0.7154	0.7207	0.9346	0.8138
$SVCFullAlign$	0.7119	0.7229	0.9197	0.8095
$SVCUnnormalizedAlign + BOW$	0.7246	0.7345	0.9171	0.8157
$SVCNormalizedAlign + BOW$	0.7316	0.7459	0.9040	0.8176
$SVCFullAlign + BOW$	0.7344	0.7457	0.9110	0.8201
Finch et al. [32]	0.7496	0.7658	0.8980	0.8266
Liu et al. [64]	0.736	0.745	0.916	0.822
STS [47]	0.7260	0.7470	0.8910	0.8130
Qiu et al. [81]	0.720	0.725	0.934	0.816
Lexico-Syntactic Subsumption [88]	0.7061	0.7207	0.9111	0.8048
Mihalcea et al. [68]	0.703	0.696	0.977	0.813
OMIOTIS [98]	0.6997	0.7078	0.9340	0.8052

CHAPTER 7

DISCUSSION

Recall from Chapter 1 that I have sought the answers to six questions. Each question will be revisited here to see what has been learned, and what still remains uncertain.

First, given a number of corpus-based and knowledge-based methods as previously proposed for word and text semantic similarity, what are the measures that work best for the task of short answer grading? While there are a number of word and text similarity measures that have been proposed in the past, no previous work has considered a comprehensive evaluation of all the measures for the task of short answer grading. I have filled this gap by running comparative evaluations of several knowledge-based and corpus-based measures on a data set of short student answers. The results (see Section 5.1) indicate that when used without feedback, the scores obtained with the best knowledge-based measures (Lesk, Jiang & Conrath, and Hirst & St. Onge) outperform the best corpus-based measures (latent semantic analysis [LSA] and explicit semantic analysis [ESA]) in my recent experiments by a significant margin[1] However, it should not be ignored that corpus-based approaches benefit crucially from their language independence. It is much easier to create a large domain-sensitive corpus than an equally beneficial language knowledge base (e.g. WordNet).

Second, given a corpus-based measure of similarity, what is the impact of the domain and the size of the corpus on the accuracy of the measure? In my early work (see Section 4.2.1), it was found that significant improvements can be obtained for the LSA measure when using a medium-sized, domain-specific corpus built from Wikipedia compared to a larger, full Wikipedia corpus. This indicates that noise contributes to LSA error in a significant way. In fact, when using LSA, the results suggest that the corpus domain may be much more important than corpus size once a certain threshold size has been reached. These experiments also clearly show that ESA works best when left alone, using all of the Wikipedia articles

[1]I suspect that the forced change in training set contributed to the much poorer quality of the ESA and LSA metrics for the MM2011 evaluations compared to those performed on MM2009. Further work should be done to confirm this.

available as dimensions.

Third, to what extent is it possible to enhance the quality of the grading system by supplementing the gold-standard answer with the answers of other students? It can be seen from these experiments (see Section 4.2.2) that there may be some advantage to automatically enhancing the instructor answer using other student answers. The greatest improvement in Pearson's correlation was 0.047 (or 10%). The limiting factor in this case was the small number of student answers to select from. I believe this warrants further study – particularly if a large (e.g. greater than 100 answers per question) short answer dataset is ever made public for researchers.

Fourth, does the dependency parse structure of a text provide clues that can be exploited to improve upon existing bag-of-words (BOW) methodologies for short answer grading? It would seem that the rudimentary alignment features introduced here (see Section 5.3) are not sufficient to act as a stand-alone grading system. However, even with a very primitive attempt at alignment detection, I show that it is possible to improve upon grade learning systems that only consider BOW features. The correlations associated with the hybrid systems (esp. those using normalized alignment data) frequently show an improvement over the BOW-only support vector machine (SVM) systems. This is true for each SVM system when considering either the correlation or the error metric.

Fifth, to what extent can machine learning be leveraged to improve upon existing approaches to short answer grading? The SVM learning techniques used in this work are clearly able to utilize multiple BOW measures to yield improvements over individual BOW metrics (see Section 5.6). For example, the correlation for the BOW-only SVM model for SVMRank improved upon the best BOW feature from .462 to .480. Likewise, using the BOW-only SVM model for support vector regression (SVR) reduces the root mean squared error by .022 overall compared to the best BOW feature.

Finally, can the methodologies I have proposed for short answer grading be successfully used to detect textual similarity, paraphrase, and entailment? Excluding the textual entailment task where the results of this methodology were far below state-of-the-art systems, all of my experiments seem to suggest that SVM regression using knowledge-based and corpus-based measures as features is a viable tactic, and more often than not adding alignment-based features produces an improved system. Perhaps surprising is that in many cases (esp. Li30 and MSRP), the knowledge-based measures alone perform admirably which is further validation of the insights of our group's earlier work in similarity [68].

BIBLIOGRAPHY

[1] E. Agirre, D. Cer, M. Diab, and A. Gonzalez-Agirre, *Semeval-2012 task 6: A pilot on semantic textual similarity*, In Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics (*SEM 2012)., 2012.

[2] V. Aleven, O. Popescu, and K.R. Koedinger, *A tutorial dialogue system with knowledge-based understanding and classification of student explanations*, Working Notes of 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Citeseer, 2001.

[3] I.E. Allen and J. Seaman, *Going the distance: Online education in the united states, 2011.*, Sloan Consortium (2011), 44.

[4] J.R. Anderson, A.T. Corbett, K.R. Koedinger, and Ray. Pelletier, *Cognitive tutors: Lessons learned*, The journal of the learning sciences 4 (1995), no. 2, 167–207.

[5] I. Androutsopoulos and P. Malakasiotis, *A survey of paraphrasing and textual entailment methods*, Arxiv preprint arXiv:0912.3747 (2009).

[6] E. Arnott, P. Hastings, and D. Allbritton, *Research methods tutor: Evaluation of a dialogue-based tutoring system in the classroom*, Behavior Research Methods 40 (2008), no. 3, 694–698.

[7] F.G. Ashby and N.A. Perrin, *Toward a unified theory of similarity and recognition.*, Psychological review 95 (1988), no. 1, 124.

[8] G. Aston and L. Burnard, *The bnc handbook: exploring the british national corpus with sara*, Edinburgh Univ Pr, 1998.

[9] Y. Attali and J. Burstein, *Automated essay scoring with e-rater v. 2*, Journal of Technology, Learning, and Assessment 4 (2006), no. 3, 1–31.

[10] Leo L. Azure, *Test grading machine*, 1966.

[11] L.F. Bachman, N. Carr, G. Kamei, M. Kim, M.J. Pan, C. Salvador, and Y. Sawaki, *A reliable approach to automatic assessment of short answer free responses*, 2002, pp. 1–4.

[12] C. Banea, S. Hassan, M. Mohler, and R. Mihalcea, *UNT: A supervised synergistic approach to semantic text similarity*, 2012.

[13] Daniel Bär, Torsten Zesch, and Iryna Gurevych, *A reflective view on text similarity*, Proceedings of the International Conference Recent Advances in Natural Language Processing 2011 (Hissar, Bulgaria), RANLP 2011 Organising Committee, September 2011, pp. 515–520.

[14] S. Brin and L. Page, *The anatomy of a large-scale hypertextual web search engine*, Computer networks and ISDN systems 30 (1998), no. 1-7, 107–117.

[15] S.M. Brookhart, *Teachers' grading practices: Meaning and values*, Journal of Educational Measurement 30 (1993), no. 2, 123–142.

[16] J. Burstein, *The e-rater® scoring engine: Automated essay scoring with natural language processing.*, (2003).

[17] J. Burstein, S. Wolff, and C. Lu, *Using lexical semantic techniques to classify free-responses*, Breadth and depth of semantic lexicons (1999), 1–18.

[18] P.G. Butcher and S.E. Jordan, *A comparison of human and computer marking of short free-text student responses*, Computers & Education 55 (2010), no. 2, 489–499.

[19] D. Callear, J. Jerrams-Smith, and V. Soh, *CAA of Short Non-MCQ Answers*, Proceedings of the 5th International Computer Assisted Assessment conference (2001).

[20] N. Chambers, D. Cer, T. Grenager, D. Hall, C. Kiddon, B. MacCartney, M.C. de Marneffe, D. Ramage, E. Yeh, and C.D. Manning, *Learning alignments and leveraging natural logic*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 165–170.

[21] W. Chu and S.S. Keerthi, *New approaches to support vector ordinal regression*, Proceedings of the 22nd international conference on Machine learning, ACM, 2005, pp. 145–152.

[22] P.A. Cohen, J.A. Kulik, and C.L.C. Kulik, *Educational outcomes of tutoring: A meta-analysis of findings*, American educational research journal 19 (1982), no. 2, 237.

[23] M. Collins, *Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms*, Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-02) (Philadelphia, PA), July 2002.

[24] L.H. Cross and R.B. Frary, *Hodgepodge grading: Endorsed by students and teachers alike*, Applied Measurement in Education 12 (1999), no. 1, 53–72.

[25] I. Dagan, O. Glickman, and B. Magnini, *The pascal recognising textual entailment challenge*, Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment (2006), 177–190.

[26] M.C. de Marneffe, T. Grenager, B. MacCartney, D. Cer, D. Ramage, C. Kiddon, and C.D. Manning, *Aligning semantic graphs for textual inference and machine reading*, Proceedings of the AAAI Spring Symposium, Citeseer, 2007.

[27] M.C. de Marneffe, B. MacCartney, and C.D. Manning, *Generating typed dependency parses from phrase structure parses*, LREC 2006, 2006.

[28] M.C. De Marneffe, A.N. Rafferty, and C.D. Manning, *Finding contradictions in text*, Proceedings of ACL-08: HLT (2008), 1039–1047.

[29] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman, *Indexing by latent semantic analysis*, Journal of the American society for information science 41 (1990), no. 6, 391–407.

[30] W.B. Dolan and C. Brockett, *Automatically constructing a corpus of sentential paraphrases*, Proc. of IWP, 2005.

[31] C. Fellbaum, *Wordnet, an electronic lexical database*, The MIT Press, 1998.

[32] A. Finch, Y.S. Hwang, and E. Sumita, *Using machine translation evaluation techniques to determine sentence-level semantic equivalence*, Proceedings of the Third International Workshop on Paraphrasing (IWP2005), 2005, pp. 17–24.

[33] Y. Freund and R. Schapire, *Large margin classification using the perceptron algorithm*, Machine Learning 37 (1999), 277–296.

[34] E. Gabrilovich and S. Markovitch, *Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge*, Proceedings of the National Conference on Artificial Intelligence, vol. 21, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006, p. 1301.

[35] ______, *Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis*, Proceedings of the 20th International Joint Conference on Artificial Intelligence (2007), 6–12.

[36] L.R. Gay, *The comparative effects of multiple-choice versus short-answer tests on retention*, Journal of Educational Measurement 17 (1980), no. 1, 45–50.

[37] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan, *The third pascal recognizing textual entailment challenge*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 1–9.

[38] N. Goodman, *Seven strictures on similarity*, Problems and projects (1972), 437–447.

[39] A.C. Graesser, K. Wiemer-Hastings, P. Wiemer-Hastings, and R. Kreuz, *Autotutor: A simulation of a human tutor*, Cognitive Systems Research 1 (1999), no. 1, 35–51.

[40] A.D. Haghighi, A.Y. Ng, and C.D. Manning, *Robust textual inference via graph matching*, Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2005, pp. 387–394.

[41] S. Hassan and R. Mihalcea, *Semantic relatedness using salient semantic analysis*, Proceedings of AAAI Conference on Artificial Intelligence, 2011.

[42] V. Hatzivassiloglou, J. Klavans, and E. Eskin, *Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning*, Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (1999).

[43] A. Hickl and J. Bensley, *A discourse commitment-based framework for recognizing textual entailment*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 171–176.

[44] D. Higgins, J. Burstein, D. Marcu, and C. Gentile, *Evaluating multiple aspects of coherence in student essays*, Proc. of HLT-NAACL, 2004.

[45] G. Hirst and D. St-Onge, *Lexical chains as representations of context for the detection and correction of malapropism*, WordNet: An Electronic Lexical Database (Christiane Fellbaum, ed.), MIT Press, 1998, pp. 305–332.

[46] T. Hughes and D. Ramage, *Lexical semantic relatedness with random graph walks*, Proceedings of EMNLP-CoNLL, 2007, pp. 581–589.

[47] A. Islam and D. Inkpen, *Semantic text similarity using corpus-based word similarity and string similarity*, (2008).

[48] J.J. Jiang and D.W. Conrath, *Semantic similarity based on corpus statistics and lexical taxonomy*, Arxiv preprint cmp-lg/9709008 (1997).

[49] T. Joachims, *Optimizing search engines using clickthrough data*, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2002, pp. 133–142.

[50] K.S. Jones, *A statistical interpretation of term specificity and its application in retrieval*, Journal of documentation 28 (1972), no. 1, 11–21.

[51] S. Jordan and T. Mitchell, *e-assessment for learning? the potential of short-answer free-text questions with tailored feedback*, British Journal of Educational Technology 40 (2009), no. 2, 371–385.

[52] A. Kennedy and S. Szpakowicz, *Evaluating rogetâĂŹs thesauri*, Paragraph 10244 (2008), 6443.

[53] T.K. Landauer and S.T. Dumais, *A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.*, Psychological review 104 (1997), no. 2, 211.

[54] C. Leacock and M. Chodorow, *Combining local context and wordnet similarity for word sense identification*, WordNet: An electronic lexical database 49 (1998), no. 2, 265–283.

[55] ———, *C-rater: Automated Scoring of Short-Answer Questions*, Computers and the Humanities 37 (2003), no. 4, 389–405.

[56] M.D. Lee, B. Pincombe, and M. Welsh, *An empirical evaluation of models of text document similarity*, Proceedings of the 27th Annual Conference of the Cognitive Science Society (2005), 1254–1259.

[57] M. Lesk, *Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone*, Proceedings of the 5th annual international conference on Systems documentation, ACM, 1986, pp. 24–26.

[58] B. Levin, *English verb classes and alternations: A preliminary investigation*, vol. 348, University of Chicago press Chicago, IL, 1993.

[59] L. Li, Y. Zhou, B. Yuan, J. Wang, and X. Hu, *Sentence similarity measurement based on shallow parsing*, Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on, vol. 7, IEEE, 2009, pp. 487–491.

[60] Y. Li, D. McLean, Z.A. Bandar, J.D. O'Shea, and K. Crockett, *Sentence similarity based on semantic nets and corpus statistics*, IEEE Transactions on Knowledge and Data Engineering (2006), 1138–1150.

[61] D. Lin, *An information-theoretic definition of similarity*, Proceedings of the 15th international conference on machine learning, vol. 1, San Francisco, 1998, pp. 296–304.

[62] M. Lintean, C. Moldovan, V. Rus, and D. McNamara, *The role of local and global weighting in assessing the semantic similarity of texts using latent semantic analysis*, Twenty-Third International FLAIRS Conference, 2010.

[63] X. Liu, Y. Zhou, and R. Zheng, *Sentence similarity based on dynamic time warping*, Semantic Computing, 2007. ICSC 2007. International Conference on, IEEE, 2007, pp. 250–256.

[64] X.Y. Liu, Y.M. Zhou, and R.S. Zheng, *Measuring semantic similarity within sentences*,

Machine Learning and Cybernetics, 2008 International Conference on, vol. 5, IEEE, 2008, pp. 2558–2562.

[65] B. MacCartney, T. Grenager, M.C. de Marneffe, D. Cer, and C.D. Manning, *Learning to recognize features of valid textual entailments*, Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Association for Computational Linguistics, 2006, p. 48.

[66] B. MacCartney and C.D. Manning, *Natural logic for textual inference*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 193–200.

[67] J. Mathews, *Just whose idea was all this testing*, The Washington Post 14 (2006).

[68] R. Mihalcea, C. Corley, and C. Strapparava, *Corpus-based and knowledge-based approaches to text semantic similarity*, Proceedings of the American Association for Artificial Intelligence (AAAI 2006) (Boston), 2006.

[69] T. Mitchel, *E-assessment of short answer questions. a white paper*, 2006.

[70] T. Mitchell, T. Russell, P. Broomhead, and N. Aldridge, *Towards robust computerised marking of free-text responses*, Proceedings of the 6th International Computer Assisted Assessment (CAA) Conference (2002).

[71] M. Mohler, R. Bunescu, and R. Mihalcea, *Learning to grade short answer questions using semantic similarity measures and dependency graph alignments*, Proceedings of the Association for Computational Linguistics – Human Language Technologies (ACL-HLT 2011) (Portland, Oregon, USA), 2011.

[72] M. Mohler and R. Mihalcea, *Text-to-text semantic similarity for automatic short answer grading*, Proceedings of the European Association for Computational Linguistics (EACL 2009) (Athens, Greece), 2009.

[73] A. Moschitti, *Making tree kernels practical for natural language learning*, Proceedings of EACLâĂŹ06, 2006.

[74] D. Nicol, *E-assessment by design: using multiple-choice tests to good effect*, Journal of Further and Higher Education 31 (2007), no. 1, 53–64.

[75] R.D. Nielsen, W. Ward, and J.H. Martin, *Recognizing entailment in intelligent tutoring systems*, Natural Language Engineering 15 (2009), no. 04, 479–501.

[76] E.B. Page, *The imminence of... grading essays by computer*, The Phi Delta Kappan 47 (1966), no. 5, 238–243.

[77] S. Patwardhan, S. Banerjee, and T. Pedersen, *Using measures of semantic relatedness for word sense disambiguation*, Computational Linguistics and Intelligent Text Processing (2003), 241–257.

[78] M.T. Pazienza, M. Pennacchiotti, and F.M. Zanzotto, *Textual entailment as syntactic graph distance: a rule based and a svm based approach*, Proceedings of the recognizing textual entaiment challenge workshop, 2005, pp. 11–13.

[79] T. Pedersen, S. Patwardhan, and J. Michelizzi, *WordNet:: Similarity-Measuring the Relatedness of Concepts*, Proceedings of the National Conference on Artificial Intelligence (2004), 1024–1025.

[80] S.G. Pulman and J.Z. Sukkarieh, *Automatic Short Answer Marking*, ACL WS Bldg Ed Apps using NLP (2005).

[81] L. Qiu, M.Y. Kan, and T.S. Chua, *Paraphrase recognition via dissimilarity significance classification*, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2006, pp. 18–26.

[82] R. Raina, A. Haghighi, C. Cox, J. Finkel, J. Michels, K. Toutanova, B. MacCartney, M.C. de Marneffe, C.D. Manning, and A.Y. Ng, *Robust textual inference using diverse knowledge sources*, Recognizing Textual Entailment (2005), 57.

[83] P. Resnik, *Using information content to evaluate semantic similarity in a taxonomy*, Arxiv preprint cmp-lg/9511007 (1995).

[84] J.J. Rocchio, *Relevance feedback in information retrieval*, (1971).

[85] C.P. Rosé, A. Roque, D. Bhembe, and K. Vanlehn, *A hybrid text classification approach for analysis of student essays*, Proceedings of the HLT-NAACL 03 workshop on Build-

ing educational applications using natural language processing-Volume 2, Association for Computational Linguistics, 2003, pp. 68–75.

[86] H. Rubenstein and J.B. Goodenough, *Contextual correlates of synonymy*, Communications of the ACM 8 (1965), no. 10, 627–633.

[87] V. Rus, A. Graesser, and K. Desai, *Lexico-syntactic subsumption for textual entailment*, Recent Advances in Natural Language Processing IV: Selected Papers from RANLP 2005 (2007), 187.

[88] V. Rus, P.M. McCarthy, A.C. Graesser, and D.S. McNamara, *Identification of sentence-to-sentence relations using a textual entailer*, Research on Language & Computation 7 (2009), no. 2, 209–229.

[89] G. Salton, *Automatic text processing: The transformation, analysis, and retrieval of*, Addison-Wesley, 1989.

[90] G. Salton, A. Wong, and C.S. Yang, *A vector space model for automatic indexing*, Communications of the ACM 18 (1975), no. 11, 613–620.

[91] J. Sinclair et al., *Collins cobuild english dictionary for advanced learners*, vol. 3rd ed., Harper Collins, 2001.

[92] A.J. Smola and B. Schölkopf, *A tutorial on support vector regression*, Statistics and computing 14 (2004), no. 3, 199–222.

[93] C. Stray, *The shift from oral to written examination: Cambridge and oxford 1700–1900*, Assessment in Education: Principles, Policy & Practice 8 (2001), no. 1, 33–50.

[94] J.Z. Sukkarieh and J. Blackmore, *c-rater: Automatic content scoring for short constructed responses*, Proceedings of the 22nd International Conference for the Florida Artificial Intelligence Research Society, Florida, USA, 2009.

[95] J.Z. Sukkarieh and S.G. Pulman, *Information extraction and machine learning: Automarking short free text responses to science questions*, Proceeding of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology, IOS Press, 2005, pp. 629–637.

[96] J.Z. Sukkarieh, S.G. Pulman, and N. Raikes, *Auto-Marking 2: An Update on the*

UCLES-Oxford University research into using Computational Linguistics to Score Short, Free Text Responses, International Association of Educational Assessment, Philadephia (2004).

[97] M. Tatu and D. Moldovan, *Cogex at rte3*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 22–27.

[98] G. Tsatsaronis, I. Varlamis, and M. Vazirgiannis, *Text relatedness based on a word thesaurus*, Journal of Artificial Intelligence Research 37 (2010), no. 1, 1–40.

[99] A. Tversky, *Features of similarity.*, Psychological review 84 (1977), no. 4, 327.

[100] K. VanLEHN, *The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems*, Educational Psychologist 46 (2011), no. 4, 197–221.

[101] K. VanLehn, P.W. Jordan, C.P. Rosé, D. Bhembe, M. Bottner, A. Gaydos, M. Makatchev, U. Pappuswamy, M. Ringenberg, A. Roque, et al., *The architecture of Why2-Atlas: A coach for qualitative physics essay writing*, Lecture Notes in Computer Science (2002), 158–167.

[102] R. Wang and G. Neumann, *Recognizing textual entailment using sentence similarity based on dependency tree skeletons*, Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, 2007, pp. 36–41.

[103] P. Wiemer-Hastings, D. Allbritton, and E. Arnott, *Rmt: A dialog-based research methods tutor with or without a head*, Intelligent Tutoring Systems, Springer, 2004, pp. 141–229.

[104] P. Wiemer-Hastings, K. Wiemer-Hastings, and A. Graesser, *Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis*, Artificial Intelligence in Education (1999), 535–542.

[105] Z. Wu and M. Palmer, *Verbs semantics and lexical selection*, Proceedings of the 32nd annual meeting on Association for Computational Linguistics, Association for Computational Linguistics, 1994, pp. 133–138.

[106] E. Yeh, D. Ramage, C.D. Manning, E. Agirre, and A. Soroa, *Wikiwalk: random walks on wikipedia for semantic relatedness*, Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing, Association for Computational Linguistics, 2009, pp. 41–49.

[107] B. Zadrozny and C. Elkan, *Transforming classifier scores into accurate multiclass probability estimates*, 2002.