7-19-2017

Measuring Semantic Textual Similarity and Automatic Answer Assessment in Dialogue Based Tutoring Systems

Rajendra Banjade

MEASURING SEMANTIC TEXTUAL SIMILARITY AND AUTOMATIC ANSWER ASSESSMENT IN DIALOGUE BASED TUTORING SYSTEMS

by

Rajendra Banjade

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Major: Computer Science

The University of Memphis

August 2017

DEDICATION

To my parents and grandparents ..

ACKNOWLEDGMENTS

I should first recognize my advisor and committee chair Dr. Vasile Rus for his great mentorship through the years. He always made his time available for thoughtful discussions and reviews. His pleasant and very supportive personality has made my life as a PhD student much better. I am also grateful to the rest of my dissertation committee members Dr. Andrew McGregor Olney, Dr. Lan Wang, and Dr. Deepak Venugopal for their helpful discussions and suggestions.

Also, I would like to acknowledge the financial supports from the grant R305A100875 from the Institute for Education Sciences to Dr. Vasile Rus, and the Institute for Intelligent Systems.

Over the years I have had affecting conversations with my friends and colleagues including Dr. Nobal Niraula, Dr. Mihai Lintean, Dr. Dan Stefanescu, Dr. Vivek Datla, Nabin Maharjan, Dipesh Gautam, Borhan Semai and the list goes on. I am grateful to them for their help and collaboration in research activities.

My special thanks goes to my wife Deepa Pandey for her great support and patience. I also remember my friends, relatives, teachers, and mentors and colleagues of the companies I worked for, who directly or indirectly helped me throughout my journey to this moment.

Rest, I owe to my parents and grand parents for their love and sacrifice. Specially to my late father who even asked his engineer boss to help buy a very nice scientific calculator for me and motivated me towards science.

ABSTRACT

Banjade, Rajendra Ph.D. The University of Memphis. August, 2017. Measuring Semantic Textual Similarity and Automatic Answer Assessment in Dialogue Based Tutoring Systems. Major Professor: Vasile Rus, Ph.D.

This dissertation presents methods and resources proposed to improve on measuring semantic textual similarity and their applications in student response understanding in dialogue based Intelligent Tutoring Systems.

In order to predict the extent of similarity between given pair of sentences, we have proposed machine learning models using dozens of features, such as the scores calculated using optimal multi-level alignment, vector based compositional semantics, and machine translation evaluation methods. Furthermore, we have proposed models towards adding an interpretation layer on top of similarity measurement systems. Our models on predicting and interpreting the semantic similarity have been the top performing systems in SemEval (a premier venue for the semantic evaluation) for the last three years. The correlations between our models' predictions and the human judgments were above 0.80 for several datasets while our models being very robust than many other top performing systems. Moreover, we have proposed Bayesian models to adapt similarity models across domains.

We have also proposed a novel Neural Network based word representation mapping approach which allows us to map the vector based representation of a word found in one model to the another model where the word representation is missing, effectively pooling together the vocabularies and corresponding representations across models. Our experiments show that the model coverage increased by few to several times depending on which model's vocabulary is taken as a reference. Also, the transformed representations were well correlated to the native target model vectors showing that the mapped representations can be used with confidence to substitute the missing word representations in the target model.

Furthermore, we have proposed methods to improve open-ended answers

assessment in dialogue based tutoring systems which is very challenging because of the variations in student answers which often are not self contained and need the contextual information (e.g., dialogue history) in order to better assess their correctness. In that, we have proposed Probabilistic Soft Logic (PSL) models augmenting semantic similarity information with other knowledge.

To detect intra- and inter-sentential negation scope and focus in tutorial dialogs, we have developed Conditional Random Fields (CRF) models. The results indicate that our approach is very effective in detecting negation scope and focus in tutorial dialogue context and can be further developed to augment the natural language understanding systems.

Additionally, we created resources (datasets, models, and tools) for fostering research in semantic similarity and student response understanding in conversational tutoring systems.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Chapter 1

Introduction

One of the earliest goals of Artificial Intelligence (AI) was to successfully emulate a human in terms of conversational ability. But how would we know the computer is thinking then? Alan Turing suggested that if the responses from the computer were indistinguishable from that of a human, the computer could be said to be thinking - often known as Turing test (Turing, 1950). And the odyssey to achieve this goal is carried on.

Towards that end, a lot of conversational systems have been developed and this is one of the active areas of research in recent years (High, 2012; J. D. Williams et al., 2015). Some popular commercial applications includes Google home, Apple's Siri, Amazon Echo, and Microsoft's Cortana. In fact, IBM has developed a tool called Watson which won the popular show Jeopardy (High, 2012). These kind of tools need Natural Language Understanding (NLU) capability which is about making computers understand our language (e.g., English and Spanish). This enables them to understand users' commands and serve accordingly. Though still young, they are very promising applications and can be considered as "killer apps" of this era.

Similarly, the natural language understanding has been used in educational domain. For example, Massive Open Online Course (MOOC) systems where thousands of students participate can benefit from automatic assignment checking (Kulkarni et al., 2015). Furthermore, computer tutors that mimic human tutors with conversational dialogue have been successfully built with the hope that a computer tutor could be available to every student with access to a computer. They are called Intelligent Tutoring Systems (ITS; Graesser, VanLehn, Rosé, Jordan, & Harter, 2001; Rus, DMello, Hu, & Graesser, 2013; VanLehn et al., 2007), the target

application of this dissertation. Some of the successful ITS systems are: AutoTutor (Graesser, Chipman, Haynes, & Olney, 2005), DeepTutor (Rus, DMello, et al., 2013), GuruTutor (Olney et al., 2012), CIRCSIM-Tutor (Evens & Michael, 2006), Why2 (VanLehn et al., 2007) and they are motivated by the effectiveness of one-on-one human tutoring (Bloom, 1984). An extensive review of tutoring research by VanLehn et al. (2007) and Rus, DMello, et al. (2013) showed that computer tutors are as effective as human tutors. It means that there is something about the one-on-one connection that is critical, whether the student communicates with humans or computers.

Though there are different tutoring systems that rely on natural language conversations and they might have been designed differently, their ultimate goal is to mimic the human tutoring in terms of conversational and pedagogical capabilities. For illustration purpose, let's take a specific example of a state-of-the-art conversational tutoring system called DeepTutor (Rus, DMello, et al., 2013). The DeepTutor's interface as shown in Figure 1.1 displays the description of a conceptual physics problem along with supporting multimedia (image) and a dialogue history. The problem which we often refer to as task is authored by the domain experts and contains the description of the task, prompts, expected/target answers to those questions, possible hints, and so on. The system asks questions and the user interacts with it in the form of natural language texts; i.e., students type their answers in sentential form and the dialogue continues. The dialogue includes multiple cycles of tutor-student interactions in the form of (1) tutor question, (2) student response, and (3) relevant feedback from the tutor until all the expectations (goal) of the task are complete which is usually the point at which student masters the concept. During this process, the system has to understand the student's response and generate appropriate feedback.

However, in order to reach towards the higher end of the successful tutoring,

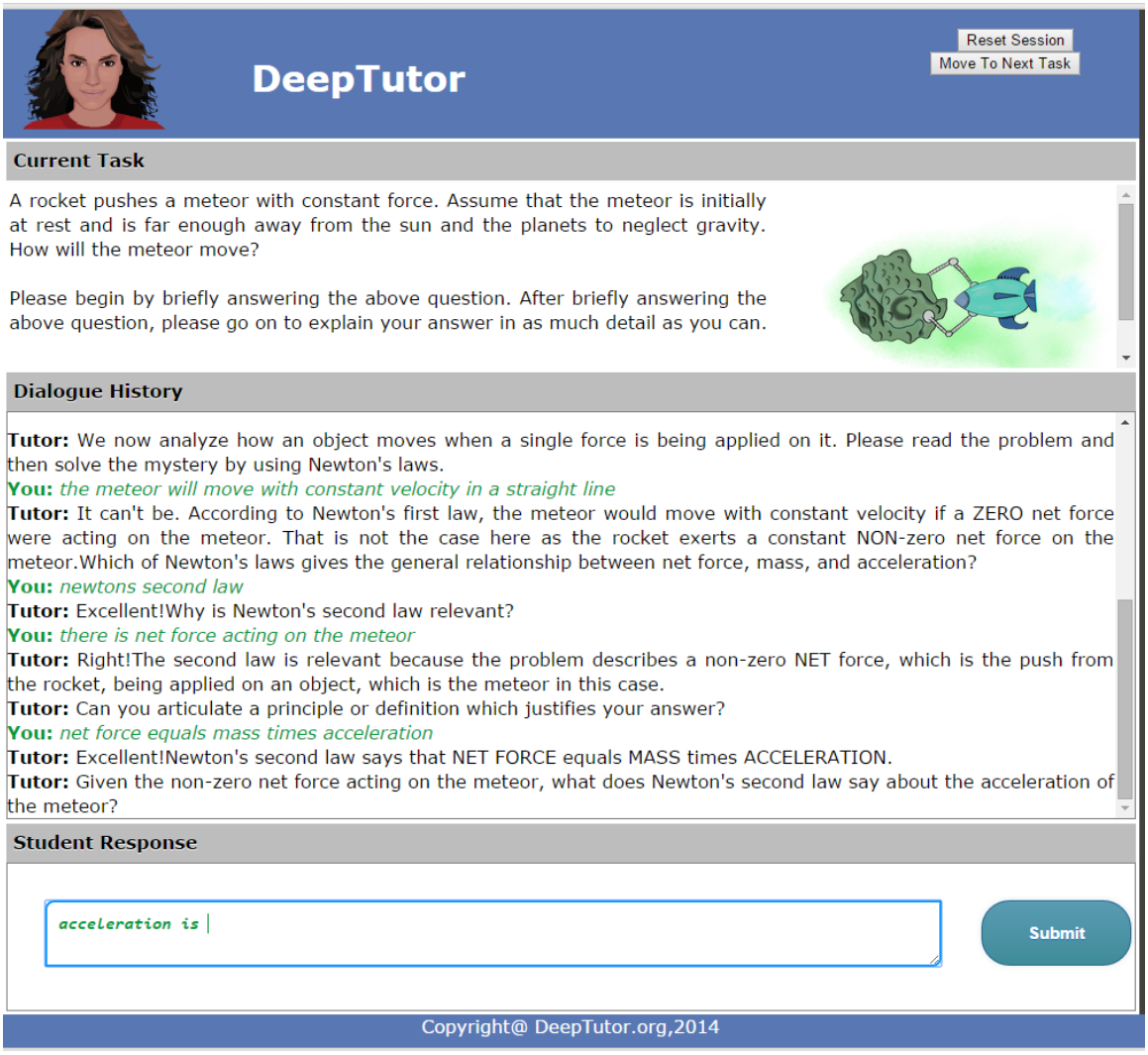


Fig. 1.1: Interface of DeepTutor tutoring system.

there are several aspects to be improved in tandem. For instance, Rus, DMello, et al. (2013) have highlighted (a) learner-tailored content and tasks, (b) effective dialogue and language processing algorithms to guide the interaction between the tutor and tutee, and (c) focusing on other aspects of learning in addition to cognitive aspects, such as affect and motivation as important factors of successful tutoring and they also found that there is much more to do in these areas.

Particularly, in the age of user-friendly interfaces, pleasant and easy interaction is an essential aspect of the design of any system. Well designed natural language dialogue systems can meet this requirement. In another words, the natural

language understanding is backbone of a conversational system, such as ITS. Therefore, the quality of these algorithms has a direct impact on core ITS tasks such as summative and diagnostic assessment, that is, the detection and tracking of students knowledge states, and providing formative feedback. One of the goals of this dissertation is to improve the learning experience by improving the interaction between the tutor and tutee through improved answer assessment models.

For student answer assessment, deep natural language understanding capability is required which is intractable as it requires collecting huge amount of knowledge (domain knowledge, world knowledge, linguistic knowledge, and contextual information) and doing inference over them. This is a yet to-be-solved problem in artificial intelligence. Alternatively, semantic similarity assessment methods have been used as a practical alternative to the true understanding approach. In that, the student answer is compared with a reference (or target) answer provided by the domain expert as illustrated in the Figure 1.2. If they are highly similar, it indicates the answer is correct. If not, depending on other factors the answer may be regarded as partially correct, incorrect, or irrelevant. But it should be noted that, measuring the similarity between texts, such as student answer and the reference answer is a much more challenging problem as discussed next.

Measuring semantic similarity between texts is to quantify the extent of their similarity in terms of their meanings. The dictionary definition of similarity is: resembling without being identical (cf. Oxford Dictionary). For example, *intelligent* and *genius* are similar words. Depending on the granularity of the texts, we can talk about the following fundamental text-to-text similarity problems: word-to-word similarity, phrase-to-phrase similarity, sentence-to-sentence similarity, paragraph-to-paragraph similarity, or document-to-document similarity. Mixed combinations are also possible such as assessing the similarity of a word to a

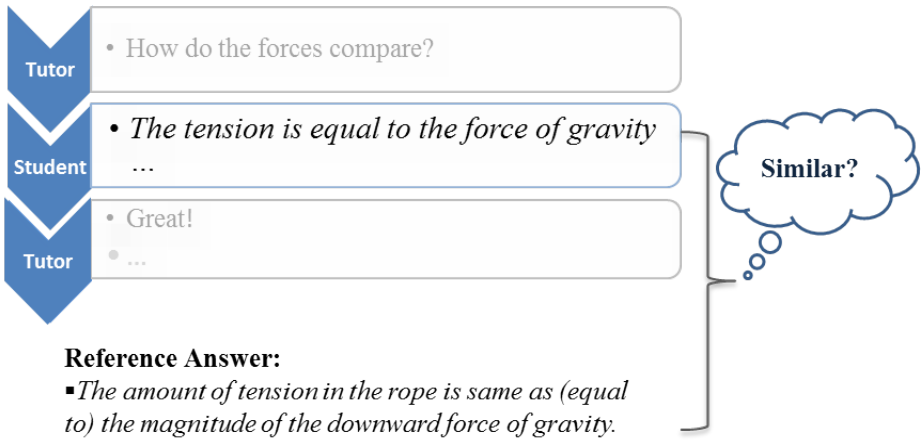


Fig. 1.2: Snippet of a dialogue showing a student answer recorded during a DeepTutor experiment and a reference answer given by the subject matter expert.

sentence or a sentence to a paragraph. For instance, in summarization it might be useful to assess how well a sentence summarizes an entire paragraph. But our goal is to use semantic similarity techniques in short answer (few words to couple of sentences) assessment which is typically done by measuring the similarity between student answer and the reference answer. In Table 1.1, we have presented a set of example sentence pairs along with similarity annotation guideline (or rubric) and the human judgment scores (Agirre et al., 2016). Each pair in the examples has been assigned a score in the range of 0 to 5. The score of 5 means the sentences are equivalent in meaning whereas 0 means they are not similar at all, and so on. Though the scores shown in the examples are full numbers, the similarity scores can be in continuous scale indicating the graded nature of similarity perceived by humans.

Because of the widespread use of semantic similarity methods, such as automatic answer grading (M. C. Lintean, Moldovan, Rus, & McNamara, 2010; Mohler & Mihalcea, 2009; Rus & Graesser, 2006), text summarization (Nenkova & McKeown, 2012), and plagiarism detection (Shrestha & Solorio, 2015), a considerable amount of effort has been put on calculating the semantic similarity or

Table 1.1: Examples of sentence similarity where similarity scores (SS) are assigned by human annotators in the scale of [0, 5] (Agirre et al., 2016).

SS	Scoring rubric (with example)
(5)	The two sentences are completely equivalent, as they mean the same thing. *The bird is bathing in the sink.* *Birdie is washing itself in the water basin*
(4)	The two sentences are mostly equivalent, but some unimportant details differ. *In May 2010, the troops attempted to invade Kabul.* *The US army invaded Kabul on May 7th last year, 2010.*
(3)	The two sentences are roughly equivalent, but some important information differs/missing. *John said he is considered a witness but not a suspect.* *"He is not a suspect anymore." John said.*
(2)	The two sentences are not equivalent, but share some details. *They flew out of the nest in groups.* *They flew into the nest together.*
(1)	The two sentences are not equivalent, but are on the same topic. *The woman is playing the violin.* *The young lady enjoys listening to the guitar.*
(0)	The two sentences are on different topics. *John went horseback riding at dawn with a whole group of friends.* *Sunrise at dawn is a magnificent view to take in if you wake up early enough for it.*

relatedness between texts (Agirre et al., 2015; Androutsopoulos & Malakasiotis, 2010; Corley & Mihalcea, 2005; Fernando & Stevenson, 2008; Landauer, Foltz, & Laham, 1998; Rus, DMello, et al., 2013). When you name a Natural Language Processing (NLP) application, a similar feature is used in one way or other. However, this long standing problem in AI has posed several challenges and it has in fact drawn a lot of attention in recent years which is also highlighted by the organization of Semantic Textual Similarity (STS) challenge as part of the semantic evaluation (SemEval; Agirre et al., 2015, 2016) program, a premier venue for the semantic evaluation and overwhelming participation for several years.

One of the problems which we have addressed is missing words in word representation models where semantics (i.e., meaning) of each word is represented in

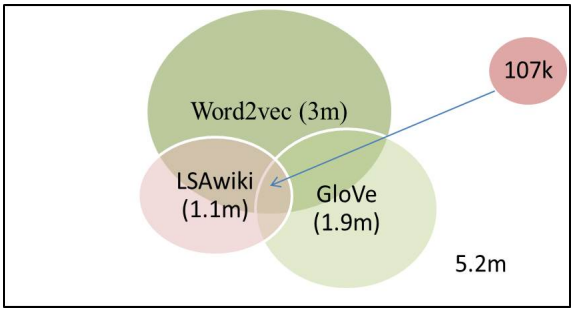


Fig. 1.3: Vocabulary size of three different pre-trained models (k - thousand, m - million).

terms of continuous vectors (also called embeddings), such as Latent Semantic Analysis (LSA; Landauer et al., 1998), Word2vec (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013), and GloVe (Pennington, Socher, & Manning, 2014). Preferably, and which is often the case, meaning representations of words are derived in an unsupervised way from extremely large collections of texts. For instance, the word2vec and GloVe word vector representations trained on texts containing billions of tokens and cover millions of unique words: the pre-trained word2vec model covers 3 million unique words, and the GloVe model has coverage of 1.9 million words. Similarly, a Latent Semantic Analysis (LSA) model developed from the whole set of Wikipedia articles (LSA_{wiki}; Stefanescu, Banjade, & Rus, 2014b) contains word representations for 1.1 million unique words.

While these are impressive numbers compared to manually created resources such as WordNet (Miller, 1995), it is interesting to observe that the previously mentioned unsupervised vector models individually cover only few million words as shown in Figure 1.4. In another words, a lot of words in the web for example are missing from each of these models. Because of missing word representations, similarity calculation methods relying on such representations are affected. Let's suppose we have to calculate similarity between two words - *deeptutor* and *tutoring* using representations from a model but it would not be possible if representation of at least one of them is not available in that model.

Additionally, a set of problems include dependent on contextual information and presence of various linguistic phenomena in the text. Though the concept of context is vague in itself, the way we describe context is in terms of two attributes: linguistic context (which is in the scope of primary field of research of this dissertation) and nonlinguistic or experiential context (which is also incorporated in our models). Linguistic context is the language that comprises the discourse which is under analysis. For instance, assessing the student response in dialogue based tutoring systems by comparing them against reference answer requires understanding the contextual information (linguistic) as illustrated by the examples in Figure 1.4. In fact, approximately 1 in every 4 answers required contextual information (e.g., previous utterance in dialogue) to properly evaluate them by the human annotators themselves (Banjade, Maharjan, Niraula, Gautam, et al., 2016). For example, pronouns used by students often refer to entities in the previous utterances, i.e., in context. Experiential contexts include such things as the type of communicative event, the topic, setting, the difficult level of questions, prior knowledge of students, etc.

As illustrated in the Figure 1.4, the student answers may vary greatly. For instance, answer $A1$ is elliptical (Carberry, 1989; Carbonell, 1983) - incomplete utterance but the meaning can be understood from the given context (dialogue history in this example). Such elliptical utterances are common in conversations even when the speakers are instructed to produce more syntactically and semantically complete utterances (Carbonell, 1983). Furthermore, the "bug" in $A2$ is referring to the mosquito and "they" in $A3$ is referring to the amount of forces exerted to each other which is also very common. In an analysis of tutorial conversation logs, Niraula et al. (2014) found that 68% of the pronouns used by students were referring to entities in the previous utterances or in the problem

Problem description: A car windshield collides with a mosquito, squashing it. **Tutor question**: How do the amounts of the force exerted on the windshield by the mosquito and the force exerted on the mosquito by the windshield compare?

Reference answer: The force exerted by the windshield on the mosquito and the force exerted by the mosquito on the windshield are an action-reaction pair.

Student answers:
A1. *Equal*
A2. *The force of the bug hitting the window is much less than the force that the window exerts on the bug*
A3. *they are equal and opposite in direction*
A4: *equal and opposite*

Fig. 1.4: A conceptual physics problem and a set of real student answers to the given question extracted from DeepTutor (Rus et al (2013)) experiment records. The dialogue context is needed to fully understand these answers.

description. In addition to anaphora, complex coreferences are also employed by students.

Similarly, there may be other linguistic phenomena present in the texts, such as Negation (Huddleston, Pullum, et al., 2002; Konstantinova, De Sousa, & Sheila, 2011; Morante & Blanco, 2012; Rooth, 1996; Wedin, 1990). A negator (or negation cue), is a lexical item that expresses negation, such as no, and not. The part of the sentence affected by the negation cue is called negation scope. The part of the scope that is most prominently negated is called negation focus (Huddleston et al., 2002). For example, the desk stops moving because [there is] <no> [{net force} acting on it] where negation cue, scope, and focus are in <>, [], and {} respectively. In fact, negation is twice as frequent in dialogue as in literary text (Tottie, 1993) and the scope can be dependent on the dialogue context. Our analysis of tutorial dialogues shows that about 9% of the student responses have some form of explicit negation (Banjade & Rus, 2016). As the negation can completely change the meaning of the text as shown in the example, the text (or dialogue) understanding systems should be able to handle them.

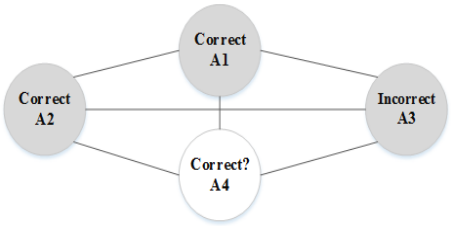


Fig. 1.5: Illustrating that we maybe able to infer the correctness of an answer based on student's performance in other related questions.

Moreover, despite the fact that the semantic similarity methods have been widely used for answer assessment and such methods are performing well in many cases, the implied assumption is that the student answers are self contained (i.e., grammatically and semantically complete and can be evaluated without needing much additional information) which is not always true. Particularly, in dialogue based tutoring systems, students might feel that they are having conversation with human tutor in chat room like environment and do not become more formal in writing responses (as shown in Figure 1.4). The off-the-shelve NLP tools (such as tools for coreference resolution) are mostly developed from general text and are not the best fit for the texts in conversational systems. That is, the error might propagate in the standard NLP pipeline making them less effective. Another approach is to augment the semantic similarity model by adding non-linguistic information. For example, as illustrated in Figure 1.5, a student giving correct answer to the most of the difficult questions will probably answer the easier question correctly. However, to the best of our knowledge, there is no such published work that combines linguistic and non-linguistic information in order to assess the open-ended answers in conversational tutoring systems. Also, there is no annotated dataset available to perform such experiments.

Furthermore, the similarity scores given by the systems are often opaque, i.e., it is difficult (if not impossible) to interpret/explain why the similarity score between given sentence pair is high or low. The interpretation of such system

output is a challenge in machine learning in general but the direct use of any system that is able to interpret the predicted similarity score is on generating diagnostic feedback. In another words, such systems will be able to answer why the student's answer was deemed incorrect, if any? Otherwise, the student would be confused and the uncertainty and cognitive load can lead to lower levels of learning as studied by (Shute, 2008). For example, *net force is zero* and *force is zero* can have different interpretations in science. If former is expected but later is the student's response, i.e., when framed in a different way the similarity score is not very high and system tells that the answer is partially correct, student might not be able to figure out what's wrong in his or her response. Though some research shows that confusion can be beneficial for learning (DMello, Lehman, Pekrun, & Graesser, 2014), it might not be true particularly when the student is not very motivated and the system does not consistently provide the correct feedback.

In brief, although a lot of progress has been made in semantic similarity and automatic answer grading research (Basu, Jacobs, & Vanderwende, 2013; Dzikovska et al., 2013; Landauer, 2003; Mohler & Mihalcea, 2009; Rus & Lintean, 2012; Sukkarieh & Blackmore, 2009), there have been several issues such as the ones mentioned before and we speculate that the research in these fields is reaching to plateau unless such issues, though difficult problems in AI, are addressed. Towards overcoming them, the work of this dissertation aims to contribute by creating methods and resources that are not available to address various research questions.

By addressing these issues in natural language understanding, the overall learning experience with the conversational tutoring systems will be enhanced and the student engagement will lead to positive outcome which can be inferred from the effectiveness of such educational tools.

Goal

The applications we target are online Intelligent Tutoring System prototypes

and the work in this dissertation is motivated by our continuous efforts to improve the semantic similarity functions and assessing the correctness of the answers given by the students interacting with such systems for which semantic similarity methods are used along with the other knowledge.

Research questions

The specific research questions which we have addressed (but not limited to) are:

- How to build improved sentence similarity measures and develop approaches to interpret the predicted similarity score by the system?
- How to cope with missing word representations (also called embeddings) in vector based word representation models?
- How to adapt the similarity systems across different domains?
- How to improve answer assessment in dialogue based intelligent tutoring systems where contextual information is important and various linguistic phenomena (e.g., coreferences, ellipsis, negation) are present?
- How to build answer assessment models including, in addition to semantic similarity features, non-linguistic knowledge, such as question difficulty and student's knowledge level?
- How can we contribute in creating resources (datasets and tools) for semantic similarity and answer assessment research?

Contributions

The contributions of this dissertation work are outlined below.

In **Chapter 2**, we present our models to measure the semantic similarity between sentences. Our sentence similarity prediction models are based on 40+ features including optimal multi-level alignment based similarity, vector based

compositional semantic methods, machine translation evaluation methods, and many others (Banjade, Niraula, et al., 2015; Banjade, Maharjan, Gautam, & Rus, 2016). In one of the approaches, we developed Support Vector Regression (SVR) models which were among the top performing systems in SemEval Short Text Similarity competitions 2015 and 2016 (Agirre et al., 2015, 2016). SemEval is a premier venue for semantic evaluation. Also, our models are more robust than many other top performing systems. In an another approach, we have treated semantic similarity probabilistically and proposed Bayesian models (C. K. Williams & Rasmussen, 1996) for predicting the distribution of sentence similarity scores for the given set of features and for domain adaptation using transfer learning approach. Those models are more intuitive to understand and expressive than frequentist counterpart, such as SVR model.

Additional contributions on developing models and resources on semantic similarity that are not presented in this dissertation are the development of composite model for measuring word-to-word similarity (Banjade, Maharjan, Niraula, Rus, & Gautam, 2015), development of SEMILAR Toolkit (a publicly available and widely used semantic similarity toolkit; Rus, Lintean, Banjade, Niraula, and Stefanescu (2013)), and the development of Latent Semantic Analysis (LSA) models from whole set of English Wikipedia articles (Stefanescu et al., 2014b). The SEMILAR toolkit and the LSA models are freely available for download for research purposes.

In **Chapter 3**, we present our novel approach of handling missing words in vector based word representation models, such as LSA (Landauer et al., 1998), GloVe (Pennington et al., 2014), and Word2vec (Mikolov, Sutskever, et al., 2013). In our approach, the vector representation of words are transformed using Neural Network (NN) models from one model where they are present (*source*) to another model where they are missing (*target*) (Banjade, Maharjan, Gautam, & Rus, 2017).

With this approach, we have significantly improved the coverage of three different types of popular pre-built models LSA, GloVe, and Word2vec. Furthermore, our intrinsic and extrinsic evaluations of transformed representations show that they can be used with confidence to substitute the missing word representations in target model. Our approach has potential to be equally applicable to phrases and sentences which are even more sparser than words. We have also made our tool (VR-Map) available for download.

In **Chapter 4**, we present our proposed models for student answer assessment tailored to dialogue based tutoring systems and the dataset we created for the evaluation of such models. One of our proposed approach for answer assessment is to apply semantic similarity model with contextual word weighting scheme. In an another approach, we have proposed building a logical and probabilistic reasoning model using Probabilistic Soft Logic (PSL; Kimmig, Bach, Broecheler, Huang, & Getoor, 2012), a version of Markov Logic Network (MLN; Richardson and Domingos (2006)). This model is capable of capturing the complex interactions among variables, such as the effect of student's performance on previous questions on the correctness of current answer. Our model including semantic similarity information along with non-linguistic information, such as student's knowledge level and question difficulty improved the accuracy by about 4% when compared to the results obtained using semantic similarity information only. An another model in which the priors are learned separately further improved the result by about 3%.

We also present an evaluation dataset we created called DT-Grade (named after DeepTutor tutoring system; Banjade, Maharjan, Niraula, Gautam, et al. (2016)) which contains 900 answers to open-ended questions recorded during students' interactions with DeepTutor tutoring system (Rus, DMello, et al., 2013) in which contextual information (i.e., previous utterances) is also very important in

understanding and assessing the answers. Each response in the dataset was annotated for: (a) it's correctness, (b) whether the contextual information was helpful in understanding the student answer, and (c) whether the student answer contained important extra information than typically expected. In fact, approximately 1 in every 4 answers required contextual information to properly evaluate them by the human annotators themselves. This type of dataset was not previously available and we have made this dataset available for research purposes.

Chapter 5 presents methods to detect negation scope and focus in tutorial dialogue (Banjade, Niraula, & Rus, 2016) and the negation dataset we created (Banjade & Rus, 2016). We collected and annotated a corpus from real dialogues between the computer tutor DeepTutor and high-school students. The corpus is called the DT-Neg corpus - DeepTutor Negation corpus (Banjade & Rus, 2016) - and consists of 1,088 instances. The corpus was manually annotated with negation cue, scope, and focus. The negations in the dataset are inter-sentential (the scope is in the sentence other than the sentence where the negation cue is present) or intra-sentential (the scope of negation is within the sentence where the cue is present). We then developed a method to detect negation scope and focus based on Conditional Random Fields (CRF) (Banjade, Niraula, & Rus, 2016). We report results for focus detection with and without use of dialogue contextual features. The results indicate that our approach is very effective in detecting negation scope and focus in tutorial dialogue context and can be further developed to augment the natural language understanding systems including the student response evaluation systems.

Chapter 6 presents our approach towards building interpretable semantic similarity models with the target of generating diagnostic feedback by answer assessment models. While useful, the quantitative or even qualitative assessments are hard to interpret because they do not provide details, i.e., they do not explain or

justify why the similarity score was assigned high or low. One way to provide an explanatory layer to text similarity assessment methods is to align chunks (sometimes referred as phrases, e.g. noun phrase) between texts and assigning semantic relation to each alignment. Our system relies on specific rules and similarity function and aligns the chunks across sentences, assigns semantic labels (e.g., EQUI - chunks are equivalent, REL - related, SPE1/2 - chunk in first/second sentence is specific than in second/first), and also predicts the similarity scores for the alignment quantifying the strength of similarity between the aligned chunks. Overall, our systems consistently performed the best in interpretable similarity challenge in SemEval 2015 and 2016 (Agirre et al., 2015, 2016; Banjade, Niraula, et al., 2015; Banjade, Maharjan, Niraula, & Rus, 2016).

Finally, in **Chapter 7** we conclude this dissertation with future directions.

Chapter 2

Measuring Short Text Similarity

Measuring semantic similarity is to quantify the extent of similarity in the meanings of two given texts. The Table 1.1 shows examples of pairs of sentences along with their average similarity scores assigned by human annotators. As discussed in Chapter 1, a considerable amount of effort has been put on calculating the semantic similarity or relatedness between texts (see Section 2.1 for the details). Our goal is to use semantic similarity techniques in short answer assessment (see Chapter 5). Therefore, we focus on the more specific task of measuring the similarity of short texts, i.e., quantifying to what extent the given two words/sentences are similar in meaning. In this chapter, we present the Support Vector Regression (SVR) models and Bayesian models we have proposed to measure the semantic similarity. The pipeline of our system is outlined in Figure 2.1. The preprocessing and feature generation steps are common to all of our models.

Our SVR models for sentence similarity use various features including the similarity scores obtained using optimal multi level alignments, vector based compositional semantics and other general features. We also call this system DTSim (named after DeepTutor lab; Banjade, Maharjan, Gautam, & Rus, 2016). We evaluated our sentence similarity models with SemEval 2016 evaluation data (Agirre et al., 2016) and the correlations between our model's output and the human ratings were up to 0.83 in some datasets (Agirre et al., 2016; Banjade, Maharjan, Gautam, & Rus, 2016). In fact, our system was one of the top performing systems among around 100 systems (46 teams) submitted in semantic similarity competition in SemEval 2016 (Agirre et al., 2016). Similarly, one of our systems (NeRoSim; Banjade, Niraula, et al., 2015) achieved 10^{th} position (4^{th} team) but with no significant difference with the results of top performing system in SemEval 2015

Fig. 2.1: The pipeline of components of our STS models.

competition (Agirre et al., 2015). Our models are more robust than many other top performing systems as discussed in Section 2.5.

The methods proposed over the years, often criticized as frequentist approaches, give a single point estimate (i.e., single point output) with sharp decision rules, i.e., their underlying model parameters are fixed. The implied assumption is that the samples are infinite and the data are a repeatable random sample, i.e., they have the same frequency. More specifically, giving a set of similarity feature values, the outcome of a similarity model would be same every time we use that model. Also, these methods do not allow us to use our prior knowledge to inform the model parameters. But the similarity annotated samples are limited in number which by their limited nature impose random biases. For instance, the agreement between human annotators while annotating similarity data collected for STS tasks is below 90% (Agirre et al., 2016). Therefore, we argue that semantic similarity scores are stochastic variables as opposed to fixed values. The noise and ambiguities present in the natural language texts are broader reasons to treat similarity values as stochastic variables. In that, we have also proposed similarity models using Bayesian approach (Kruschke, 2014).

In addition to other features which we discuss in Section 2.6, the Bayesian models allow us to use our prior knowledge about the data which maybe updated during the training phase. In our models we have applied transfer learning approach to adapt them across domains (we have taken sources of similarity texts as domains, such as news headlines and community forums). The domain adapted Bayesian models in which the Gaussian mean priors were learned from out of domain data

performed better than the domain specific models, particularly when the domain specific training data is small.

Furthermore, we have proposed other models also (Banjade, Maharjan, et al., 2015; Banjade, Niraula, et al., 2015; Stefanescu, Banjade, & Rus, 2014a) but we focus on the aforementioned two different approaches SVR models and Bayesian models. Moreover, the similarity score given by the model is generally opaque and it's a great challenge in interpreting the numeric score; i.e., it is often difficult to explain why the similarity score produced by a model is high or low. We have proposed a model towards interpreting the similarity scores which we present in Chapter 6.

The outline of this chapter is as follows. Next, we discuss on related work. Then we describe datasets (Section 2.2) and preprocessing steps (Section 2.3) which is followed by feature extraction (Section 2.4). These steps are common to both types of models (SVR models and Bayesian models). And then we present SVR models and the results (Section 2.5). Similarly, after SVR models we present Bayesian models and the results (Section 2.6).

2.1 Related Work

Research on semantic similarity of texts focused initially on measuring similarity between individual words. Also, several methods that are proposed for measuring the similarity of sentences rely on word-to-word similarity (e.g., word alignment based methods) while others directly work at sentence level. In this section, we first discuss about word-to-word similarity methods and then discuss on various methods proposed over the years for measuring sentence level similarity.

2.1.1 Word-to-Word Similarity

Based on the types of resources used, the methods that measure semantic relatedness or similarity (often used interchangeably) of words are broadly of two types: those relying on knowledge bases, such as WordNet (Miller, 1995), and those

that infer word associations from bigger collections of texts commonly based on word co-occurrences, called distributional/distributed methods. In the knowledge base category, WordNet based methods are quite popular (Pedersen, Patwardhan, & Michelizzi, 2004; Resnik, 1995). WordNet (Miller, 1995) lexicon groups together words that have the same meaning, i.e., synonyms, into synsets (synonymous sets). Synsets are also referred to as concepts. A group of word-to-word similarity measures were defined that use lexico-semantic information in WordNet (Lin et al., 1998; Pedersen et al., 2004; Resnik, 1995).

On the other hand, in distributional models words are represented in vector forms and similarity between them is typically calculated as cosine score. The word representations are almost always learned in unsupervised manner from huge collection of texts, such as Wikipedia articles. Distributional similarity methods (or corpus based methods) include algebraic methods, such as Latent Semantic Analysis (LSA; Landauer et al., 1998; Stefanescu et al., 2014b) and Hyperspace Analog to Language (HAL; Burgess & Lund, 1995). Stefanescu et al. (2014b) have developed probably the biggest LSA models from whole English Wikipedia articles (Spring 2013 snapshot). Rus, Niraula, and Banjade (2013) have proposed of using contribution of words across topics in Latent Dirichlet Allocation (LDA; Blei, Ng, & Jordan, 2003) model as vector representations of words and using them to measure the similarity between words. This one is a probabilistic approach. Pennington et al. (2014) have proposed a combination of algebraic and probabilistic model called GloVe. Collobert et al. (2011) obtained word representation using deep neural network. In recent days, Word2vec (Mikolov, Sutskever, et al., 2013) model in which word representations are learned using single layer feed forward neural network has drawn a lot of attention. In one study we combined the word representations obtained using different models and applied them to word-to-word similarity measurement (Niraula, Gautam, Banjade, Maharjan, & Rus, 2015).

However, latent representations of words are difficult to interpret and the number of dimensions is chosen empirically. Towards using more interpretable representation of words, Gabrilovich and Markovitch (2007) have proposed Explicit Semantic Analysis (ESA) model where Wikipedia articles are used as concepts and words present in them are represented in terms of those "explicit" concepts.

Previous methods, individually or as a combination of different methods, have yielded very good performance when it comes to measuring relatedness (Jiang & Conrath, 1997; Y. Li, Bandar, & McLean, 2003; Stefanescu, Rus, Niraula, & Banjade, 2014). The popular datasets used for evaluation of word-to-word similarity models are Sim-353 (Finkelstein et al., 2001) and TOEFL analogy dataset (Turney, 2001). However, as Hill, Reichart, and Korhonen (2014) explored, distributional similarity methods are not capturing well the true similarity between words. They also published a dataset containing 999 word pairs (called Simlex-999) with human rated similarity scores explicitly quantifying the similarity of words. For example, *lemon* and *tea* would get lower score as they are not similar despite the fact that they are related. It fosters the development of applications that benefit from similarity than those which take into account a broader range of relations. To this end, we developed a method that combines several diverse approaches that rely on corpus and knowledge bases to measure semantic similarity and achieved state-of-the-art results (Banjade, Maharjan, et al., 2015). However, similarity and relatedness are often used interchangeably. We are also using them interchangeably.

2.1.2 Sentence Level Similarity

Measuring sentence-to-sentence similarity is also a well-studied topic in NLP because of its use in many tasks such as question-answering, text mining, text summarization, plagiarism detection, assessing the correctness of student answers in education technologies, and assessing the translation quality of automatic translation systems. Due to this wide applicability, the research literature is

abundant in methods for detecting or assessing sentence similarity, which are often presented as methods for identifying paraphrases (Androutsopoulos & Malakasiotis, 2010; Corley & Mihalcea, 2005; Fernando & Stevenson, 2008; Rus & Lintean, 2012). In order to streamline and foster the research in this area, semantic similarity competitions have been organized for last several years and participation is overwhelming (Agirre, Diab, Cer, & Gonzalez-Agirre, 2012; Agirre et al., 2014, 2015, 2016).

Early applications of text similarity were in the field of information retrieval. These early developments were essentially "bag-of-words" strategies developed for solving well-known problems such as selecting the documents most relevant to a given query (Salton & Buckley, 1988) or text classification (Y. H. Li & Jain, 1998). The most basic methods rely on lexical matching (i.e., words or n-gram overlap analysis) and return scores based on the number of lexical units that occur in both fragments (e.g. sentences). Also, certain preprocessing steps such as stemming, tagging or stop-words removal were shown to improve the results of the systems. In fact, we did an analysis with different preprocessing variations and found that they can be responsible for large differences in the performance of a system (Rus, Banjade, & Lintean, 2014).

In time, the methods moved beyond lexical matching to using corpus and knowledge-based word-to-word similarity measures. However, one challenge with using word-to-word similarity measures is that they cannot be directly applied to compute similarity of larger texts such as sentences. Researchers have proposed methods to extend the word-to-word (W2W) similarity measures to text-to-text (T2T) similarity measures (Fernando & Stevenson, 2008; Han, Kashyap, Finin, Mayfield, & Weese, 2013; Rus & Lintean, 2012; Rus, Niraula, & Banjade, 2013; Sultan, Bethard, & Sumner, 2015). For instance, Rus and Lintean (2012) have applied greedy and optimal word alignment methods where they used WordNet

based word similarity methods and Latent Semantic Analysis (LSA) based methods. They also proposed quadratic alignment based method for paraphrase detection (M. Lintean & Rus, 2015). One of the models we developed is based on chunk alignment and weighting chunks by information content (Stefanescu et al., 2014a). Chunks (loosely called phrases) are more meaningful units than words. Many systems exploit machine translation evaluation measures, such as BLEU measure (Madnani, Tetreault, & Chodorow, 2012; Papineni, Roukos, Ward, & Zhu, 2002) which are generally based on word/phrase alignment. Probably, it would not be biased to say that alignment based similarity methods are the individually best performing methods despite their relative simplicity (Agirre et al., 2015, 2016; Banjade, Niraula, et al., 2015; Banjade, Maharjan, Gautam, & Rus, 2016; Rus & Lintean, 2012; Sultan et al., 2015). They are also fast to compute.

On the other hand, approaches are being developed for obtaining the representation of sentences and only then using them to find out the similarity between sentences. However, it is difficult to directly learn meaningful sentence representations because of sparseness of texts (Mikolov, Sutskever, et al., 2013; Rus, Niraula, & Banjade, 2013) and they are not the best options for measuring the similarity of short texts, such as phrases and sentences. Therefore, most of the methods use the representation of words or phrases to learn the sentence representations. One simple and commonly found effective method of combining word representation to obtain sentence representation is to add the vectors of individual words (Banjade, Niraula, et al., 2015; Sultan et al., 2015). In one of our studies, we took the weighted average of the vectors of words in each sentence by the part-of-speech category and found the consistent improvement in the results (Maharjan, Banjade, Gautam, J. Tamang, & Rus, 2017).

In recent years, deep learning techniques have been employed in learning the sentence representations from their constituents (Mikolov, Sutskever, et al., 2013;

Socher, Huang, Pennin, Manning, & Ng, 2011). For example, Socher et al. (2011) applied recursive auto encoder with dynamic pooling in order to obtain sentence representations using word representations. Once the sentence representations are learned, similarity, such as cosine between sentences can be computed easily. Most recently, Kiros et al. (2015) developed an approach using long and short term memory models for learning sentence representations in continuous vector forms, called Skip-thought vectors. Similarly, Sent2Vec tool[1] which generates sentence representations has been developed at Microsoft. It uses both Deep Structured Semantic Model (DSSM; Huang et al., 2013) and the DSSM with convolutional-pooling (CDSSSM; Gao, Deng, Gamon, He, & Pantel, 2014; Shen, He, Gao, Deng, & Mesnil, 2014) for mapping from text to low dimensional continuous vector form.

Moreover, machine learning techniques are employed to further improve results by combining various features including lexical and semantic features along with other general features, such as ratio of sentence length (Agirre et al., 2015; Banjade, Niraula, et al., 2015; Brockett & Dolan, 2005). We developed a Support Vector Regression model combining similarity scores produced from various individual methods along with other general features. Likewise, kernel based methods are also proposed (M. C. Lintean & Rus, 2011; Severyn, Nicosia, & Moschitti, 2013). Interestingly, M. C. Lintean and Rus (2011) used dissimilarity kernel to predict the paraphrase.

Providing a venue for the evaluation of state-of-the-art algorithms and models, STS shared task has been held annually since 2012 (Agirre et al., 2012, 2014, 2015, 2016). During these times, a diverse set of genres and data sources have been explored (e.g., news headlines, video and image descriptions, glosses from lexical resources including WordNet, FrameNet, OntoNotes, web discussion forums,

[1]https://www.microsoft.com/en-us/download/details.aspx?id=52365

and Q&A data sets). Dozens of teams have been participating each year and submitting results produced with their different systems. The systems including ours that combine various features including similarity from individual methods, such as alignment based and vector composition based methods have been consistently performing the best in those competitions (Agirre et al., 2015, 2016). However, many of the top performing systems were trained separately for each dataset in the evaluation data and using the training data from the same domain (Rychalska, Pakulska, Chodorowska, Walczak, & Andruszkiewicz, 2016; Sultan et al., 2015). For example, model trained using news headlines was used to assess similarity of subset of the evaluation dataset containing only the text pairs from news headlines. In another words, the models were tuned for each evaluation dataset. Our models, however, were trained using a single set of dataset and applied to full set of evaluation data. This makes our models more robust. Furthermore, we have treated semantic similarity probabilistically and proposed Bayesian models for domain adaptation using transfer learning.

In order to build and evaluate the semantic similarity (or paraphrase identification) models, various human annotated datasets are released over the years and most of them are created in recent years. We did a comprehensive study of datasets available for the research in this field (Rus et al., 2014) and here we discuss them in brief. The performance of similarity methods was (and still is) commonly evaluated using the Microsoft Research Paraphrase Corpus (MSRP; Dolan, Quirk, & Brockett, 2004). MSRP contains 5,801 sentence pairs overall, out of which 3,900 (67.23%) are considered paraphrases. However, MSRP does not seem to be the ideal data set for benchmarking similarity systems, because of the high degree of word overlap and lack of variations in that dataset. Also, each pair in the dataset is labeled either as 1 - paraphrase or 0 - non-paraphrase. This does not capture the graded meaning similarity. The SemEval organizers have released large number of

human annotated sentence pairs (more than 10 thousand pairs) over the years (Agirre et al., 2016). As opposed to MSRP corpus where each sentence pair has binary label, the sentence pairs released in SemEval competitions have 6 different labels in the range of [0 5]. Evaluation of system output is performed by comparing them with the human annotated scores and typically Pearson correlation is used for that purpose. We also used the SemEval data for model building and evaluation.

2.2 Datasets

For model building and evaluation, we used human annotated datasets released as part of several STS challenges. Models were built using datasets released as part of prior STS whereas evaluation was done using the evaluation dataset released during STS 2016. The training and evaluation/test datasets are summarized in Table 2.1 and Table 2.2 respectively. Each pair in training and test data has a human annotated similarity rating between 0 and 5 (5 means equivalent) as shown in Table 1.1. The human annotator agreement was below 90% across all datasets (Agirre et al., 2016).

Table 2.1: Summary of training data. These datasets were released over the years as part of SemEval Semantic Textual Similarity (STS) challenges.

Dataset	**Count**	**Release Year**	**Source**
SMTnews	351	STS2012-Testset	Statistical Machine Translation (SMT) evaluation
Headlines	750	STS2014-Testset	newswire headlines
Headlines	742	STS2015-Testset	newswire headlines
Deft-forum	423	STS2014-Testset	forum posts
Deft-news	299	STS2014-Testset	forum news
Answers-forums	375	STS2015-Testset	Q&A forum answers
Answers-students	750	STS2015-Testset	student answers
Belief	375	STS2015-Testset	committed belief forum texts
Total	4065		

We selected datasets that included texts from different genres. For example, STS2012-Test means that the dataset was released as a test set in Semantic Textual

Table 2.2: Summary of test data (released in STS 2016).

Dataset	**Count**	**Source**
Plagiarism	230	data from plagiarized text
Postediting	244	post edited machine translated texts
Headlines	249	newswire headlines
Question-Question	209	Stack Exchange forum questions
Answer-Answer	254	Stack Exchange forum answers
Total	1186	

Similarity (STS) competition in SemEval 2012 and so on. However, some others, such as Tweet-news were not included. The Tweet-news data were quite different from most other texts, such as they include a lot of hash tags, and we believe that they need to be handled differently.

The test data (summarized in Table 2.2) contained 1186 sentence pairs that include texts from the following sources: News Headlines, Plagiarized text, Post edited text obtained from machine translation, Question pairs taken from Stack Exchange forum, and pairs of community answers taken from Stack Exchange website. Our SVR model was developed before seeing the test set and we tuned our model using 10-fold cross validation and applied to unseen test set. On the other hand, our work on Bayesian model is mostly exploratory in nature and therefore, apart from training and test, we did not allocate separate dataset for model validation.

2.3 Preprocessing

Hyphens were replaced with whitespaces if they were not composite verbs (e.g. video-gamed). The composite verbs were detected based on the POS tag assigned by the POS tagger. Also, the words starting with co-, pre-, meta-, multi-, re-, pro-, al-, anti-, ex-, and non- were left intact. Then, the hyphen-removed texts were tokenized, lemmatized, POS-tagged and annotated with Named Entity tags using Stanford CoreNLP Toolkit (Manning et al., 2014). We also marked each word as

whether it was a stop word. We also created chunks using our own Conditional Random Fields (CRF) based chunking tool (Maharjan, Banjade, Niraula, & Rus, 2016) which outperforms OpenNLP chunker when evaluated with human annotated chunks provided in interpretable similarity shared task in 2015. We normalized texts using mapping data. For example, *pct* and *%* were changed to *percent*.

2.4 Feature Extraction

We used various features in our models including semantic similarity scores generated using individual methods and other general features, such as relative length of sentences. That is, in order to improve the robustness of our model, the similarity scores predicted using individual methods were also used as features in our final model. Before describing those individual methods, we describe word similarity methods which were also used for sentence similarity calculation.

2.4.1 Word-to-Word Similarity

We used vector based word representation models, PPDB 2.0 database (Pavlick et al., 2015), and WordNet (Miller, 1995) in order to measure the similarity between words as given below.

$$sim(w1, w2, m) = \begin{cases} 1, \text{ if w1 and w2 are synonyms} \\ 0, \text{ if w1 and w2 are antonyms} \\ \text{ppdb(w1,w2), if m = ppdb} \\ \frac{\mathbf{X1}.\mathbf{X2}}{|\mathbf{X1}||\mathbf{X2}|}, \text{ otherwise} \end{cases}$$

where $m \in \{ppdb, LSAwiki, word2vec, GloVe\}$, **X1** and **X2** are vector representations of words *w1* and *w2* respectively.

We first checked synonyms and antonyms in WordNet 3.0. If the word pair was neither synonym nor antonym, we calculated the similarity score based on the model selected. The word representation models used are: word2vec (Mikolov,

Sutskever, et al., 2013)[2], Glove (Pennington et al., 2014)[3], and LSA Wiki (Stefanescu et al., 2014b)[4]. The cosine similarity was calculated between the word representation vectors. We also used the similarity score found in PPDB database[5].

Handling missing words: We checked for the representation of a word in raw form as well as in base (lemma) form. If neither of them was found, we used vector representation of one of its synonyms in WordNet for the given POS category. The same strategy was used while using PPDB to retrieve similarity score. We have proposed an approach to better handle missing words in vector based word representation models which is presented in Chapter 3.

2.4.2 Sentence-to-Sentence Similarity

Word Alignment Based Method

In this approach, all the content words (in lemma form) in two sentences (S1 and S2) were aligned optimally (OA) using Hungarian algorithm (Kuhn, 1955) as described in (Rus & Lintean, 2012) and implemented in SEMILAR Toolkit (Rus, Lintean, et al., 2013). The process is same as finding the maximum weight matching in a weighted bipartite graph. The nodes are words and the weights are the similarity scores between the word pairs. The sentence similarity is calculated as:

$$sim(S1, S2) = 2 * \frac{\sum_{(w1,w2)\in OA} sim(w1, w2)}{|S1| + |S2|}$$

In order to avoid the noisy alignments, we reset the similarity score below 0.5 (empirically set threshold) to 0.

[2]https://code.google.com/archive/p/word2vec/

[3]http://nlp.stanford.edu/projects/glove/

[4]http://semanticsimilarity.org

[5]http://paraphrase.org/

Chunk Alignment Based Method

We chunked texts (see Section 2.3) and aligned chunks optimally as described in (Stefanescu et al., 2014a). The difference is that the chunks containing Named Entities were aligned using rules: (a) the chunks were treated as equivalent if both were named entities and at least one of the content words was matching, (b) they were treated as equivalent if one was the acronym of another. In other cases, chunk-to-chunk similarity was calculated using optimal word alignment method. The process is same as word alignment based method. First, the words in chunks were aligned to calculate chunk-to-chunk similarity. Finally, chunks in two sentences were aligned optimally for sentence level similarity. In order to avoid noisy alignments, we set similarity score to 0 below 0.5 for word alignment and 0.6 for chunk alignment. These thresholds were set empirically.

Interpretable Similarity Based Method

We aligned chunks from one sentence to another and assigned semantic relations and similarity scores for each alignment. The semantic labels were EQUI, OPPO, SIMI, REL, SPE1, SPE2, and NOALI. For example, the semantic relation EQUI was assigned if the given two chunks were equivalent. The similarity score range from 0 (no similarity) to 5 (equivalent). We aligned chunks and assigned labels as described in (Banjade, Maharjan, Gautam, & Rus, 2016; Banjade, Niraula, et al., 2015; Maharjan et al., 2016). Once the chunks were aligned and semantic relation types and similarity scores were assigned, sentence level scores were calculated for each relation type as well as an overall score was calculated using all alignment types as shown next.

$$Norm\_count(alignment - type) = \frac{(\#\text{ alignments with type = alignment-type})}{\text{Total \# alignments including NOALI}}$$

$$Similarity(S1, S2) = \frac{\sum_{(c1,c2) \in Alignments} \text{sim(c1, c2)}}{5 * (\text{Total \# alignments including NOALI})}$$

Where $c1 \in$ {S1 chunks}, $c2 \in$ {S2 chunks}, and alignment-type $\in \{EQUI, OPPO, SIMI, REL, SPE1, SPE2, NOALI\}$.

Vector Composition Based Method

In this approach, we combined vector based word representations to obtain sentence level representations through vector algebra as shown below. In a different experiment (Maharjan et al., 2017), we also weighted word vectors by their POS categories and found that giving different weights for four major POS categories (Noun, Verb, Adverb, and Adjective) consistently improved results across datasets.

$$\mathbf{RV}(S) = \sum_{w \in W} \mathbf{V}_w$$

where W is the set of content words in sentence S and V_w is the vector representation for word w. The cosine similarity was calculated between the resultant vectors - RV(S1) and RV(S2). Word representations from LSA_{wiki}, word2vec and GloVe models were used.

Similarity Matrix Based Method

The approach is similar to the word alignment based method and similarity scores for all pairs of words from given two sentences are calculated. However, a key difference is that all word-to-word similarities are taken into account, not just the maximally aligned word similarities as in (Fernando & Stevenson, 2008).

2.4.3 Feature List

All or subset of the following features was used for three different runs as described in Section 2.5. We used word2vec representation and WordNet antonym and synonym for word similarity unless anything else is mentioned specifically.

1. Similarity scores generated using word alignment based methods where word-to-word similarity was calculated using methods described in Section 2.4.1.

2. Similarity score using optimal alignment of chunks where word-to-word similarity scores were calculated using representation from word2vec model.

3. Similarity scores using similarity matrix based methods. The similarities between words were calculated using different word similarity methods discussed in Section 2.4.1.

4. Similarity scores using chunk alignment types and alignment scores (interpretable features).

5. Similarity scores using the resultant vector based method using word representations from word2vec, GloVe, and LSA Wiki models.

6. Noun-Noun, Adjective-Adjective, Adverb-Adverb, and Verb-Verb similarity scores and similarity score for other types of words using word alignment based method.

7. Multiplication of noun-noun similarity scores and verb-verb similarity scores.

8. $\frac{|C_{i1}-C_{i2}|}{C_{i1}+C_{i2}}$ where C_{i1} and C_{i2} are the counts of $i \in \{$all tokens, adjectives, adverbs, nouns, and verbs$\}$ for sentence 1 and 2 respectively.

9. Presence of adjectives and adverbs in first sentence, and in the second sentence.

10. Unigram overlap with synonym check, bigram overlap and BLEU score.

11. Number of EQUI, OPPO, REL, SIMI, and SPE relations in aligning chunks between texts relative to the total number of alignments.

12. Presence of antonym pair among all word pairs between given two sentences.

2.5 SVR Model

In this section, we describe our system SVR model, in which we used various features in order to predict the similarity score for the given sentence pairs. The features of the model are described in Section 2.4. The pipeline of components in SVR is shown in Figure 2.1.

Models and Runs. Using the combination of features described in Section 2.4.3, we built three different Support Vector Regression (SVR) models corresponding to three runs (Run1-3) submitted in SemEval 2016. In Run 1, all of the features except chunk alignment based features were used. The XL version of PPDB 2.0 was used. In Run 2, we selected the features using Weka's correlation based feature selection tool (Hall & Smith, 1998) which also included chunk alignment based similarity score. In Run 3, we took the representative features from all of the features described in Section 2.4.3. For example, alignment based similarity scores generated using word2vec model were selected as it performed relatively better in training set compared to GloVe and LSA Wiki models. Also, we used XXXL version of the PPDB 2.0 database (the precision maybe lower but the coverage is higher as compared to the smaller version of the database).

We used LibSVM library (Chang & Lin, 2011) in Weka 3.6.8[6] to develop SVR models. We evaluated our models in training data using 10-fold cross validation approach. The correlation scores in training set were 0.791, 0.773 and 0.800 for Run1, Run2, and Run3 respectively. The best results in training set was obtained using RBF kernel. All other parameters were set to Weka's default.

2.5.1 Results

Table 2.3 shows the correlation (Pearson) of our system outputs with human ratings. The graph in Figure 2.2 also includes the results of best of the best among the submitted runs as part of SemEval 2016. The correlation scores of all three runs

[6]http://www.cs.waikato.ac.nz/ml/weka/

Table 2.3: Results of our SVR model with different runs on STS 2016 test data. The number of records of each dataset in the test set was used as weight while calculating weighted correlation score.

Data set	**Run1**	**Run2**	**Run3**
Headlines	0.815	0.795	0.812
Plagiarism	0.837	0.828	0.832
Postediting	0.823	0.815	0.815
Question-Question	0.614	0.608	0.591
Answer-Answer	0.578	0.550	0.562
Weighted Mean	0.735	0.720	0.724

of our system were 0.8 or above for three datasets - Headlines, Plagiarism, and Postediting. However, the correlations are comparatively lower for Question-question and Answer-answer datasets. One of the reasons is that these two datasets are quite different from the texts we used for the training (we could not include them as such type of datasets were not available during model building). For example, the question pair (#24 in Question-question dataset): *How to select a workout plan?* and *How to create a workout plan?* have high lexical overlap but they are asking very different things. Analyzing the focus of the questions may be needed in order to distinguish the questions, i.e., the similarity between such pairs may need to be modeled differently. With the release of this type of dataset will foster the development of similarity models where the text pair consists of questions.

However, it should be noted that many of the top performing systems were trained separately for each dataset in the evaluation data and used the training data from the same domain (Rychalska et al., 2016; Sultan et al., 2015). For example, model trained using news headlines was used to assess similarity of subset of the evaluation dataset containing only the text pairs from news headlines. In another words, the models were tuned for each evaluation dataset. Our models, however, were trained using a single set of dataset and applied to full set of evaluation data. This makes our models more robust.

Another interesting observation is that the results of three different runs are

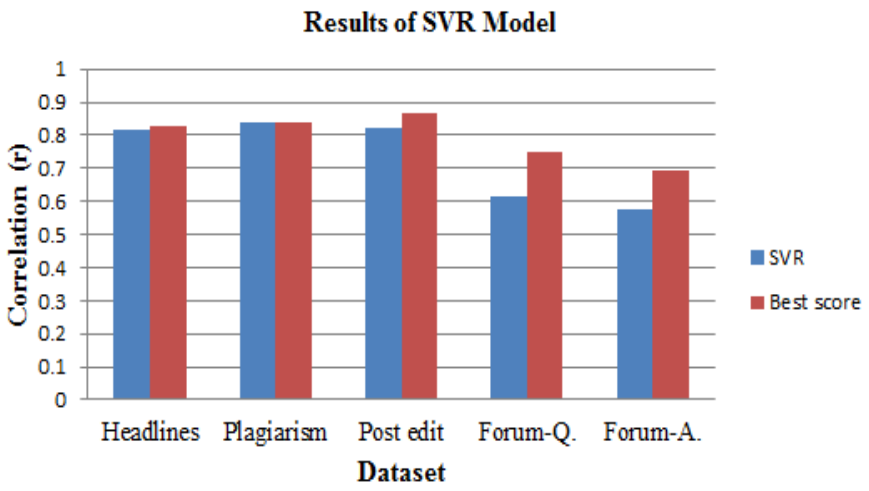


Fig. 2.2: Results of SVR model (Run1) compared to best of the best results in STS 2016 (Agirre et al., 2016).

similar to each other. The most predictive feature was the word alignment based similarity using word2vec model. The correlation in full training set was 0.725. It is not surprising considering that the alignment based systems were the top performing systems in the past shared tasks as well (Agirre et al., 2015; Han et al., 2013; Sultan et al., 2015).

2.6 Bayesian Models and Transfer Learning

As discussed earlier, we treat semantic similarity probabilistically and propose to develop Bayesian models. In order to illustrate and further motivate the development of Bayesian models for similarity, we generated density plots of human annotated similarity scores (gold scores) corresponding to different values of system predicted scores. For this purpose, we developed a Linear Regression (LR) model with several features using a set of datasets described in Table 2.1. We performed 10-fold cross validation of LR model and predicted the scores on the same dataset to see how the human annotated scores would be distributed corresponding to the scores predicted by a well fitted model. Also, the training set consists of a large number of samples (around 4,000) and selecting subsets of data corresponding to a

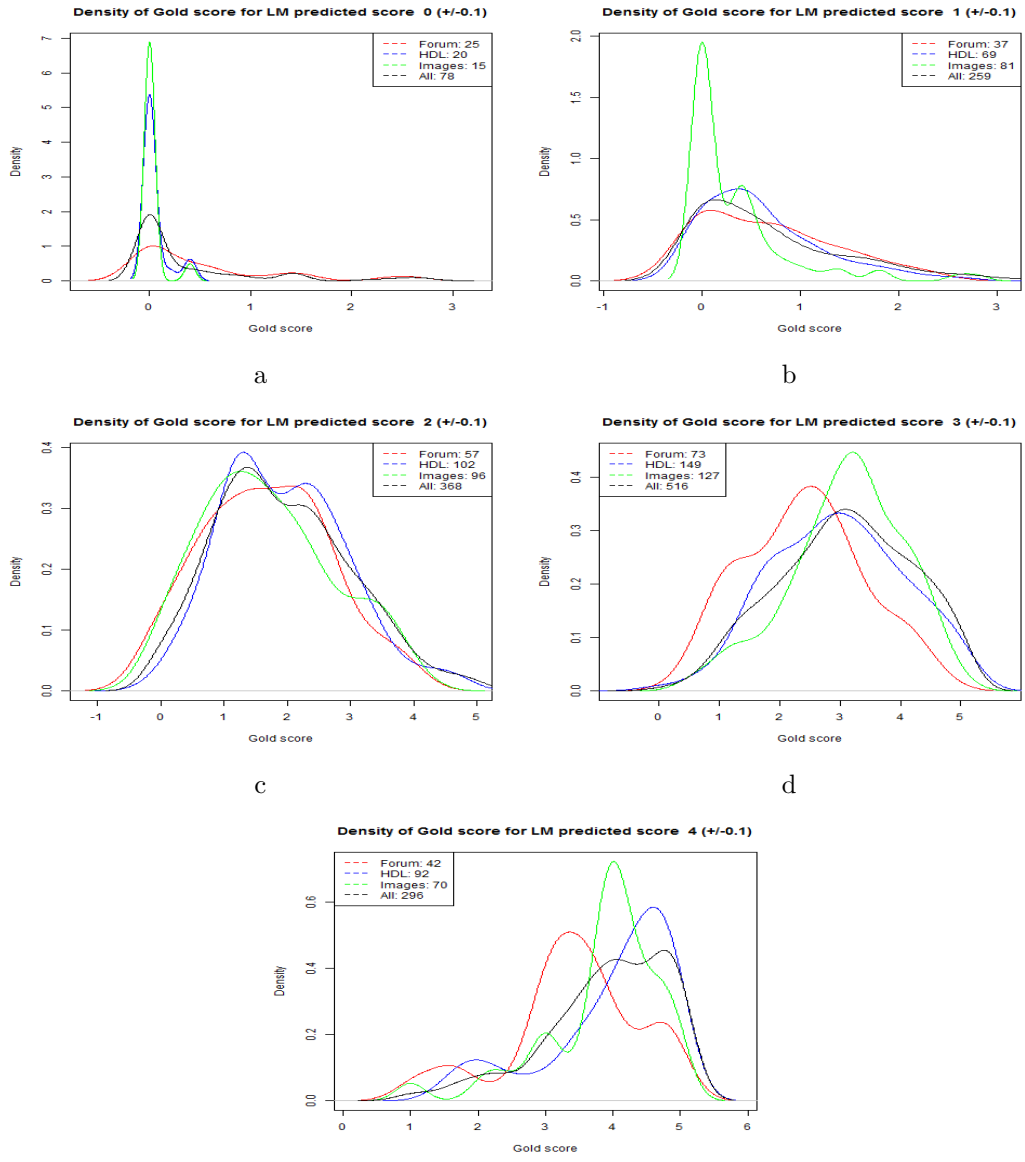


Fig. 2.3: Density plots of gold scores corresponding to selected predicted scores (selected from 0 to 4 in the interval of 1) obtained using a Linear Regression model in the training data presented in Table 2.1, for the whole dataset as well as for the different groups (domains).

predicted score and from a specific data source is possible. We then plotted the density of gold scores corresponding to selected system generated scores from 0 to 4 in the interval of 1. They are shown in Figure 2.3. We have not shown graph for LR predicted score of 5.0 due to lack of sufficient samples to generate a density plot corresponding to that score, i.e., system predicted score.

Since only some of the system predicted scores are the exact full numbers (e.g., 0, 1, 2), the predicted similarity scores within the interval of +/-0.1 from an integer/full number are rounded to that nearest integer (e.g., 0.99 was changed to 1.0, 4.05 was changed to 4.0, and so on) such that we have more samples to plot the graphs. It should be noted that the predicted similarity scores (and the human rated scores) are in the range of 0 to 5 and we considered +/-0.1 as a reasonable interval to round the predicted scores.

From Figure 2.3, we can see that there are multiple human rated scores corresponding to a single LR system generated similarity score which in turn corresponds to a set of feature values. In another words, there are different human annotated scores (considered as estimates of true similarity scores) for the same value of system generated score. Furthermore, the shape of some density plots of gold scores for different subsets (images, headlines, forums) are quite different. For example, when system predicted score is 3.0, their graphs are less overlapped and their means vary approximately from 2 to 3. These illustrate that the output of the similarity system should be a distribution and the underlying distribution from which the model parameters are derived can be different across different datasets. We considered these subsets as different domains (described in Section 2.6.2).

We cautiously modeled the output of our model as normal distribution around the mean of the predicted scores corresponding to an input x as illustrated in Figure 2.4. In this figure, X is a set of feature values, y' is the predicted value and y is the expected (gold or human annotated) score. Ideally the gold scores and

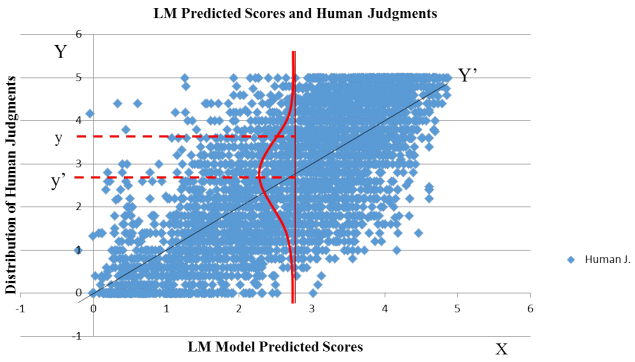


Fig. 2.4: Illustrating errors in the linear model estimates of semantic similarity. X represents the predicted score which corresponds to a set of feature values, y' is the estimated similarity score and y is the expected score (human annotated score).

the predicted scores should be same but practically that is not the case illustrated in Figure 2.4 and density plots in Figure 2.3. For better visualization, the density plots in Figure 2.3 maybe be rotated 90 degree anti-clockwise and overlaid in Figure 2.4 and the mean be positioned in the vertical line. Gold scores vary even for the same value of predicted score (y') and sometimes the mean of gold scores does not match with the predicted score by linear regression model showing the greater difference in model output and the human annotated scores. For example, in Figure 2.3b, the system output (y') is 1.0 (+/- 0.1) but the mean of gold score is clearly less than 1.0 for the full dataset as well as for the subsets.

But during the prediction time, we do not know the gold scores and the only thing we can do is to estimate the similarity scores using the model (using the LR model, for example) with the hope that the predicted scores will be as close as possible to human judgments (i.e., gold scores). Therefore, for practical reasons we considered the predicted score by LR model as the mean of our output distribution. That is, for example, when LR score is 1.0, the output will be normally distributed around the mean with some deviation (discussed later). Furthermore, the shape of all the density plots do not necessarily look same and are in some cases deteriorated from the shape of a normal distribution, but for mathematical convenience we have

modeled the output as normal distribution in all cases. On the other hand, the frequentist methods would give us the fixed score. Therefore, it makes sense to have the outcome of our system as distributions.

Bayesian models (Kruschke, 2014; C. K. Williams & Rasmussen, 1996) treat model parameters as unknown quantities and describe parameters as well as predictions probabilistically. It is the data (i.e., observation) which are fixed. Furthermore, Bayesian approaches allow us to set our belief about a domain and update the prior belief as we observe the world. Therefore, we have proposed Bayesian models for semantic similarity. The key point here is that we do not get a single point estimate, e.g., "a line of best fit", as in the frequentist case. Instead we get a distribution for each input, i.e., set of features and we set that distribution to be Gaussian distribution.

In that, we have built domain general models and domain adapted models using transfer learning (Pan & Yang, 2010). In domain general models all available training data is used to build the model and applied directly to the data in the test set. Furthermore, we posit that the domain specific models can take benefit from other domains or domain agnostic models. While the notion of domain is not very crispy, we differentiate the sources of sentence similarity text pairs as different domains (see Table 2.1 in Section 2.2). For example, some pairs come from news headlines while some others are from community forums and image annotations. We can see that headlines texts contain more named entities and are more formal than texts from community forums. As discussed above, the density plots for different types of data are also quite different from each other (see Figure 2.3) in some cases. Therefore, we treat headlines, images, and forum texts as three different domains. In the proposed transfer learning approach for the domain adaptation, the priors are learned only from the out of domain data and are later updated based on the

observations from the target domain. We describe the models in further details in Section 2.6.2.

2.6.1 Domain General Model

In this section, we discuss the domain general Bayesian model to STS. That is, a model developed from the full set of training data is applied to all the data in test set. In another words, the model is not tuned for a particular set of data. The Bayesian models proposed for domain adaptation are presented subsequently. Here we first discuss the least-square regression model which is a frequentist counter-part of a linear Bayesian model.

When given a set of training data (x_i, y_i),..., (x_n, y_n), least-square regression finds a relationship between response variable (y) and explanatory variables (x).

The least-squares regression is expressed as the following equation.

$$f(X) = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \epsilon \tag{2.1}$$

where β is the coefficient vector and k is the number of features or dimensions. The assumption in conventional linear regression is that the error ϵ (also known as measurement error) is normally distributed around mean 0; i.e., $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Then the goal is to find out the best coefficients β. It means minimizing some form of error. The most popular method to do this is sum square error (SSE), which is the sum of the squared differences between the outputs and the linear regression estimates.

$$\begin{aligned} SSE(\beta) &= \sum_{i=1}^{N} (Y_i - f(X_i))^2 \\ &= \sum_{i=1}^{N} (Y_i - \beta^T X_i)^2 \end{aligned}$$

Therefore, the goal is to minimize SSE. The Maximum Likelihood Estimate (MLE) of β is minimized when,

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

We use those β values obtain predicted values of given x. However, the β are point estimates (i.e., each take a single value). On the other hand, a linear Bayesian model (C. K. Williams & Rasmussen, 1996) represents Y and the parameters β in the form of probability distributions (p.d.f. in short).

If our hypothesis is,

$$Y' = \beta_0 + \beta_1 X_1 + ... + \beta_n X_n \tag{2.2}$$

$$Y \sim p.d.f.(Y') \tag{2.3}$$

We set the output as a Gaussian distribution with mean Y' as mentioned earlier.

$Y \sim \mathcal{N}(\mu = Y', \sigma^2)$ and the parameters β_0, β_1,..,β_n, σ are also stochastic variables each following a certain distribution, such as Gaussian distribution. We use the human annotated score for the given sentence pair as an expected similarity score (Y).

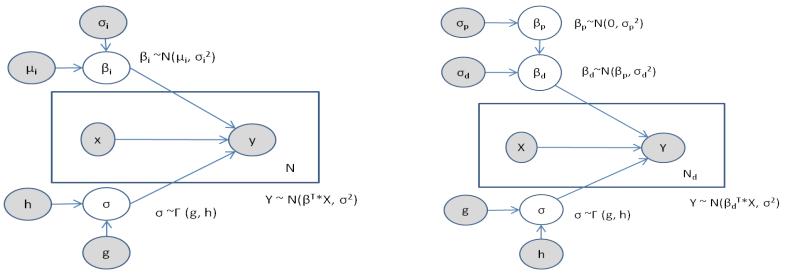
Below are some equations that further illustrate our approach. We start with Bayes rule which indicates that

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)} \tag{2.4}$$

where H is the hypothesis and D is the data (or observation). In terms of model parameters,

$$P(\beta_0, \beta_1, ..., \beta_n, \sigma|D) = \frac{\mathcal{N}(D|\beta_0, \beta_1, ..., \beta_n, \sigma)P(\beta_0, \beta_1, ..., \beta_n, \sigma)}{P(D)}$$

Assuming model parameters are independent of each other, the joint probability can be expressed as the multiplication of their probability distributions.



a b

Fig. 2.5: Graphical representation of: (a) Domain general model, (b) domain adaptation model using transfer learning. The observed variables and the hyperpriors are shaded (β - vector of model coefficients, N number of training samples, d - domain, p - prior, $\mathcal{N}$ - Normal distribution, B - Beta distribution).

$$P(\beta_0, \beta_1, ..., \beta_n, \sigma|D) = \frac{\mathcal{N}(D|\beta_0, \beta_1, ..., \beta_n, \sigma)P(\beta_0)P(\beta_1)P(...)P(\beta_n)P(\sigma)}{P(D)}$$

Furthermore, by assuming that the coefficients are from the normal distribution and the standard deviation of the output distribution follows gamma distribution,

$$P(\beta_0, \beta_1, ..., \beta_n, \sigma|D) = \frac{\mathcal{N}(D|\beta_0, \beta_1, ..., \beta_n, \sigma)\mathcal{N}(D|\beta_0, \sigma_0)(D|...)(D|\beta_n, \sigma_n)\Gamma(g, h)}{P(D)} \tag{2.5}$$

The gamma distribution (Γ) was chosen to model the standard deviation of the output distribution (σ) as it is a natural choice to sample a positive value and it is also a conjugate prior distribution for various types of distributions, such as Poisson distribution.

Then, for the new input x, the output is

$$y \sim \mathcal{N}(\mu = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n, \sigma^2) \tag{2.6}$$

Figure 2.5a shows the graphical representation of domain general model, in which the output (y) is semantic similarity score whose underlying distribution is Gaussian distribution with $\beta^T x$ mean for a set of features x $(x_1, x_2, .., x_n)$ as input (see Section 2.4.3) where n is the number of features in the input data. This model is

built with full set of training examples (N count), i.e., training set includes data from all available domains (see Table 2.1). In another words, the model is domain agnostic. The model coefficients (β) are set to Gaussian distributions with zero mean prior and each coefficient is set to have σ_i ($0 \leq i \leq n$) standard deviation. The zero mean Gaussian priors for co-efficients can also be seen as regularization of model parameters as it tends to control the parameter values from growing too large or reducing too small.

We have represented standard deviation (or variance) in terms of precision. The precision is the inverse of variance as shown below. But it should be noted that we still use the symbol σ to represent precision as an alias of variance for the ease of understanding them.

$$precision = \frac{1}{\sigma^2} \tag{2.7}$$

The representation of standard deviation (or variance) in terms of precision is influenced from the conventions used by widely used BUGS family of languages (WinBUGS, OpenBUGS and JAGS) which are used for statistical modeling (see Section 2.6.4). They apparently use "precision" in specifying distributions. We also used OpenBUGS (Lunn, Spiegelhalter, Thomas, & Best, 2009) for our experiments.

The precision for model output distribution (which is set to Gaussian distribution) is Γ distribution with pre-defined parameters g and h. The parameters g and h (used for σ) are tunable parameters. We chose positive value for the precision and which in turn gives positive value for standard deviation σ. As a natural choice, we used gamma distribution for this.

The model learning process consists of optimizing the log conditional likelihood of the data with respect to the parameters ($L(D|\beta, \sigma)$). This likelihood function can take different forms depending on the priors for the model parameters and may not have a close form solution. Thus, those likelihood optimization

parameters are estimated by using MCMC based sampling, such as Gibbs sampling. The training data consists of human annotated similarity scores which are in the range of 0 to 5 (see Section 2.2).

The domain adaptation model (Figure 2.5b) is described next.

2.6.2 Domain Adaptation using Transfer Learning

As discussed earlier, the notion of domain can be different in different context. We treat different sources of similarity text pairs as separate domains. For example, news headlines, texts from community forums, image annotations, etc. The idea of domain adaptation is that the domains have commonalities and differences, therefore, one may be able to use the information from another domain. The proposed domain adaptation model is illustrated in Figure 2.5b.

Our domain adaptation models are based on transfer learning approach (Pan & Yang, 2010). Instead of using zero mean Gaussian parameters, the priors learned from the other domain are used to initialize domain specific model co-efficients as illustrated in Figure 2.5b. We refer such priors as *informed priors*. The mean of β_d (i.e., domain model parameters) is initialized with β_p (i.e., prior parameters learned from other domain). It allows domain specific model to tune its parameters using the data from its own domain while some information of other domains might be still useful for predicting similarity scores of the domain of interest. For instance, models build using community forum texts can be adapted to news headlines by initializing the model parameters of headline model (domain model) with that learned from forum texts. This approach is more appealing when the domain specific training data is scarce (because if trained with limited samples the domain specific model can underfit) or when the target domain appears to be very new or perceived to be very different from the dataset used to train the model. Partly, the domain adaptation can also act as regularization by controlling the growth of the parameter values of the domain model.

More formally, in Domain Adaptation model (DAM) or Transfer Learning Model (TLM) the prior parameters (β_p) are learned using out of domain data only. For example, β_p are learned using Forum data only and are used as Gaussian mean priors for Headlines model. This would give us an idea of whether the out of domain information (prior) help for the target domain.

We developed three different types of models and from which first two are used for comparison purposes.

- **Out of Domain Model (ODM)**: Model with parameters learned using out of domain data only. The zero mean Gaussian priors are used to initialize model coefficients.
- **Uninformative Prior Domain Model (UPDM)**: The model parameters are learned from domain data but initialized with zero mean Gaussian prior.
- **Informed Prior Domain Model (IPDM)**: Model is developed from domain specific data only but this time the Gaussian mean priors for coefficients are learned from out of domain data. That is, IPDM model coefficients are initialized using weights learned from ODM.

2.6.3 Evaluation Methods

The Bayesian approach is more expressive and intuitive to understand when compared with frequentist methods but it is very subjective in evaluation as output of a Bayesian model is a distribution rather than a single point estimate which is easier to compare with human judgments and quantify the results, such as calculating correlation scores. Also, the existing datasets have a single value of similarity score assigned by human annotators (most often, the average of similarity scores assigned by several annotators is published) for each sentence pair. Therefore, in order to validate our models, we generated point estimates for similarity scores (as expected values) by sampling from the output distribution of

similarity scores which we set to be a Gaussian distribution. The similarity score for a given input x is calculated as:

$$\mu_i = sample_i(\beta_0) + sample_i(\beta_1)x_1 + ... + sample_i(\beta_n)x_n \tag{2.8}$$

$$score(x) = \frac{1}{N_s}\sum_{i=1}^{N_s} sample(\mathcal{N}(\mu_i, sample(\sigma)^2)) \tag{2.9}$$

where N_s - number of samples to use for score calculation. We discard the burn-in samples and use the most recent samples as they tend to be more stable. We discuss on this later.

The $score(x)$ is compared with human judgment score for the record corresponding to input x. In specific, we calculate the Pearson (r) correlation between system output and the human rated scores. To validate our model, we compared the results of our model with the results obtained using standard least-square regression. However, it should be noted that the main difference is in the interpretation of the output rather than the quantity.

Furthermore, in domain adaptation models (see Section 2.6.2), the results obtained using Informed Prior Domain Model (IPDMs) were compared with two different systems: (a) Out of Domain Model (ODM), and (b) Uninformed Prior Domain Model (UPDM). The results of ODM model tells that how well the model developed from another domain works for the domain of interest and comparing IPDM with UPDM tells the benefits of transfer learning, i.e., the benefits of using informed priors.

2.6.4 Experiments and Results

Feature Selection

Instead of working with all the features we selected the most predictive features using Weka's correlation based feature subsection selection tool (CfsSubsetEval; Hall & Smith, 1998). We chose only 6 most predictive features.

Also, reducing the number of features made Bayesian models run faster. The selected features include alignment based similarity and resultant vector based similarity features among others.

Statistical Modeling Tool - OpenBUGS

We used one of the widely used statistical modeling tools called OpenBUGS (Lunn et al., 2009) to develop our Bayesian models. It is one of the versions of BUGS software package and is useful for statical modeling, such as performing Bayesian inference using Gibbs sampling. It allows users to specify a statistical model, of (almost) arbitrary complexity, by simply stating the relationships between related variables. The software includes an 'expert system' which determines an appropriate MCMC (Markov chain Monte Carlo) scheme (e.g. the Gibbs sampling) for learning the specified model.

Results of Domain General Models

We built a domain agnostic model using all data in the training set presented in Table 2.1 and evaluated in the test set presented in Table 2.2. The model coefficients were Gaussian distribution with $\mu = 0.0$ and prior for standard deviation of output distribution was set to gamma distribution: $\Gamma(g = 0.01, h = 0.01)$. The number of iterations MCMC performed during model building by OpenBUGS tool was set to 10,000 with 5,000 burn in steps. The thin step was 1, i.e., after burn in all samples were used.

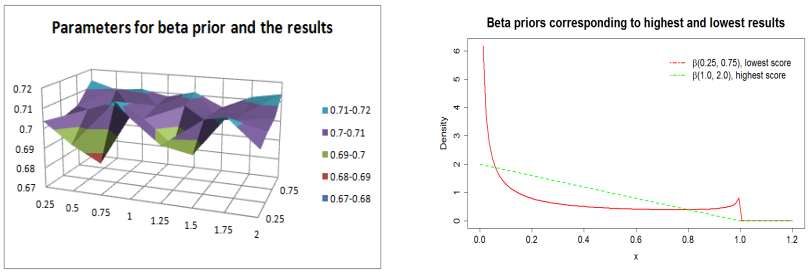
Table 2.4: Results of our domain general Bayesian models with different configurations of model coefficients (β). The results which are better than LR model results are in bold (B - Beta distribution, σ - precision).

Config	**Headlines**	**Plagiarism**	**Post Edit.**	**Q-Q**	**A-A**	**W.Avg**
LR	0.812	0.832	0.815	0.591	0.562	0.725
$\sigma = 0.04$	0.782	0.814	**0.846**	**0.603**	**0.566**	0.724
$\sigma = 1$*exp-6	0.781	0.814	0.846	0.601	0.565	0.723
B(0.1, 2)	0.727	0.773	0.813	0.597	0.513	0.685

The results are presented in Table 2.4. We have reported results of different datasets in terms of correlation between system output and human assigned similarity scores (see Section 2.6.3) as well as the weighted average of their correlations. The number of sentence pairs in each subset in evaluation data was used as weight. For comparison purpose, we also present the results obtained using Least Square Regression (LSR). Our Bayesian models are linear in nature and LSR is the frequentist counterpart of our models. We can see that the weighted average result of our model (M1) with zero mean Gaussian prior for co-efficients is 0.724 and is comparable with LSR results. In the first place, this validates that our Bayesian implementation is correct.

Furthermore, the results of subset data (Post editing, Question-Question, and Answer-Answer) are better than linear regression while the overall results are comparable. Particularly, the results of our models (that used Gaussian distribution for the parameters) for Post edited text are both 0.846 whereas it is 0.815 for LSR. We also changed the prior parameters to get some sense of whether different priors yield very different results. For instance, we changed model parameters to $\beta(a, b)$ instead of using Gaussian distributions. We found that the results are less good in the later case. The parameters of the prior distributions (e.g., a and b in $\beta(a, b)$) were chosen empirically and only few results are accommodated in the table and sensitivity of priors is analyzed in further details in Section 2.6.4. However, the varying results with different priors suggest that some prior distributions happen to be closer to the distributions that describes the underlying hidden parameters of the domain and have great importance. Based on the results, the zero-mean Gaussian prior which we chose initially has best described our model coefficients.

Next, we further analyze the sensitivity of the priors in the model outcome.



a b

Fig. 2.6: (a) Graph showing changing results (w. average correlation scores in the test set) depending on shape of beta prior distributions (i.e., changing a and b in $\beta(a, b)$) for model coefficients, (b) Shape of beta priors corresponding to the highest and lowest results on (a).

Analyzing Model Sensitivity to Priors

In Table 2.4, we have presented results with few different priors: some are zero mean Gaussian and a Beta distributions. Additionally, we chose Beta priors for model parameters and performed experiments for further analysis on the sensitivity (or effect) of model parameter priors in the model outcome. We chose beta priors as they are more expressive in terms of representing different shape of data distributions and we changed the shape of Beta distributions by varying values of a and b in $B(a, b)$ both in the interval of 0.25. The results are plotted in Figure 2.6a.

Figure 2.6a shows the results in test set with various values of a and b of the beta priors used for model coefficients. The best result (avg. score $= 0.714$) among them were obtained with $B(a = 1.0, b = 2.0)$ The lowest results were obtained with $B(a = 0.25, b = 0.75)$. The difference in highest score and the lowest result was 4 correlation points. On the right (in Figure 2.6b), we have plotted the shape of Beta distributions corresponding to highest and lowest results presented in (a). In both cases, the densities of parameters are high in the lower range (< 0.1) particularly the one which belongs to lowest results indicating that the model coefficients need to be shifted towards zero but not too harshly. These results in addition to the results presented in Table 2.4 show that the model outcome can vary greatly

depending on the priors we choose. But at the same time it enables us to use the prior information which is one of the coveted attributes of Bayesian models.

MCMC Convergence Diagnostics

Although it is virtually impossible to certainly know whether the MCMC (Markov Chain Monte Carlo) process was converged, we have run diagnostics tests that are considered good indicators of convergence. In that, we calculated gelman-rubin convergence factors (also known as shrink factor or potential scale reduction factor; Gelman & Rubin, 1992) and at the same time also observed some traceplots with different number of iterations for MCMC sampling (Gibbs sampling). The gelman-rubin factor (GR factor in short) looks at the variance of samples within the chain and across chains to calculate a single score and the factor less than 1.1 is considered as a good indicator of convergence of MCMC sampling, i.e., approximate convergence is diagnosed when the upper limit of shrink factor is close to 1. Calculating gelman-rubin factor can be automated whereas the traceplots help us visualizing the stability of the sampling distribution.

The Figure 2.7 shows samples of traceplots and shrink factor plots with 2,000 iterations and 10,000 iterations of MCMC sampling of a model parameter. We used samples from 3 chains for shrink factor calculation. In the figures we can see that the samples were not very stable with 2,000 iterations and the shrink factor also did not descend to 1.1 or lower. On the other hand, we obtained shrink factors less than 1.1 when number of iterations were set to above 5,000. Also, the traceplots show more stable samples being produced towards the end indicating the convergence of sampling. We repeated these tests with some other parameters and found the similar pattern. Therefore, we set number of iterations to be at least 10,000 in all of our experiments.

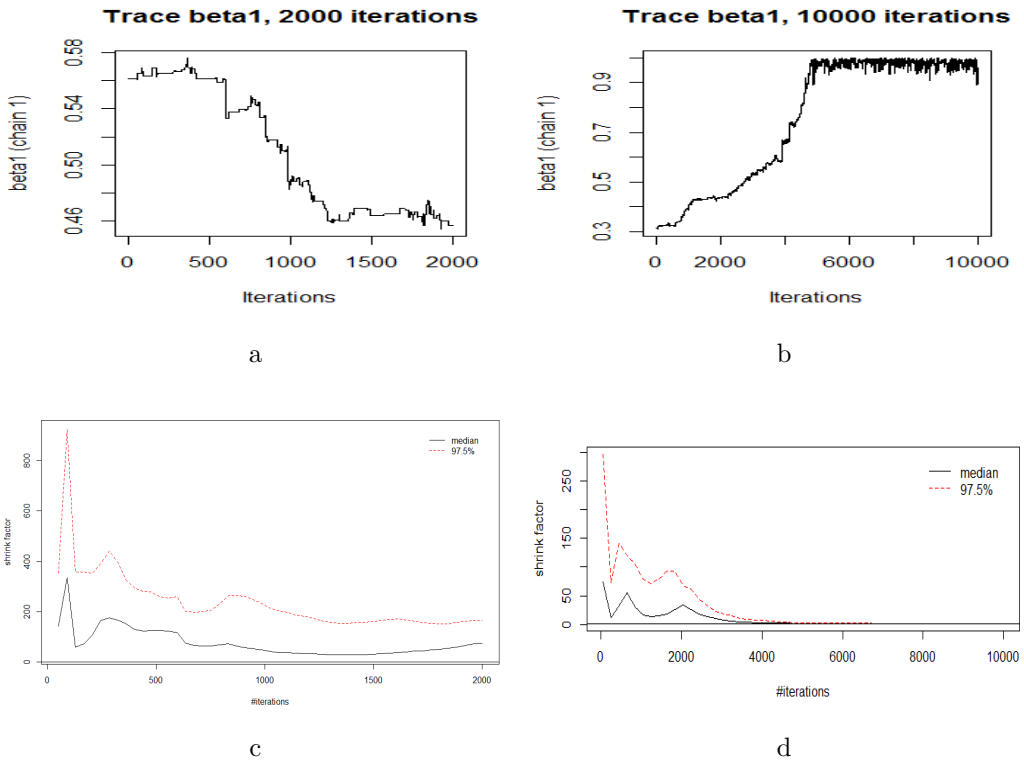


Fig. 2.7: Trace plots (top) and gelman-rubin convergence factor plots (bottom) with 2,000 iterations (left) and 10,000 iterations (right) for a model parameter.

Results of Transfer Learning Models

In order to study domain adaptation using transfer learning, we selected data from three domains: news headlines, community forum texts, and image annotations. The former two are selected from Table 2.1 and Table 2.2. The later one (image annotations) was added particularly for this study. We selected these domains for our study as the number of records in each type was quite enough for splitting into training and test. Once we collected the data, we shuffled the records in each group and split each of them into two sets - training and test. In each case, we allocated 1,000 records for training and the rest of the data was allocated for the evaluation. The equal number of training data was used for uniformity and fairness of comparisons. These datasets are summarized in Table 2.5.

By using datasets from three different domains as shown in Table 2.5, we had

Table 2.5: The datasets used for domain adaptation using transfer learning.

Domain	Training	Test	Description
Headlines (HDL)	1000	492	Headlines from training set (in Table 1)
Forum (FRM)	1000	427	Deft-forum, Belief, Answers-forums, Answer-answer combined
Images (IMG)	1000	499	Data released as part of STS 2015, and STS 2014 evaluation

six different pairs of out-of-domain and in-domain settings. For example, in one setting we learned parameters from headlines and used those parameters as priors in models developed from forum data. In this scenario, the model developed from headlines is considered as out of domain model (ODM), model developed from forum data with zero mean Gaussian prior is uninformed prior domain model (UPDM), and when the model parameters were initialized using those learned from headlines as informed priors is called informed prior domain model (IPDM). In order to study the effect of size of domain specific data, we built domain models (UPDM, and IPDM) with varying number of training records (from 100 to 1000). In each case (ODM, UPDM, or IPDM), we evaluated our models on domain specific test samples which is in this example the test set of forum data presented in Table 2.5.

The results of our domain adaptation models are plotted in Figure 2.8. In each of the six domain adaptation settings of ODM and DM, we can see that results of domain models with uninformed prior (i.e., zero mean Gaussian prior) are very lower than the adapted models when the number of domain specific training data is small (<600). When domain specific training size is increased, the results are mixed. In $FRM \rightarrow HDL$ transformation (in Figure 2.8a), the results of IPDM model is better than both ODM and UPDM showing the advantage of domain adaptation even when the domain specific training size is quite big (up to 1000). In an another example of $HDL \rightarrow IMG$ (in Figure 2.8e), when the training data is

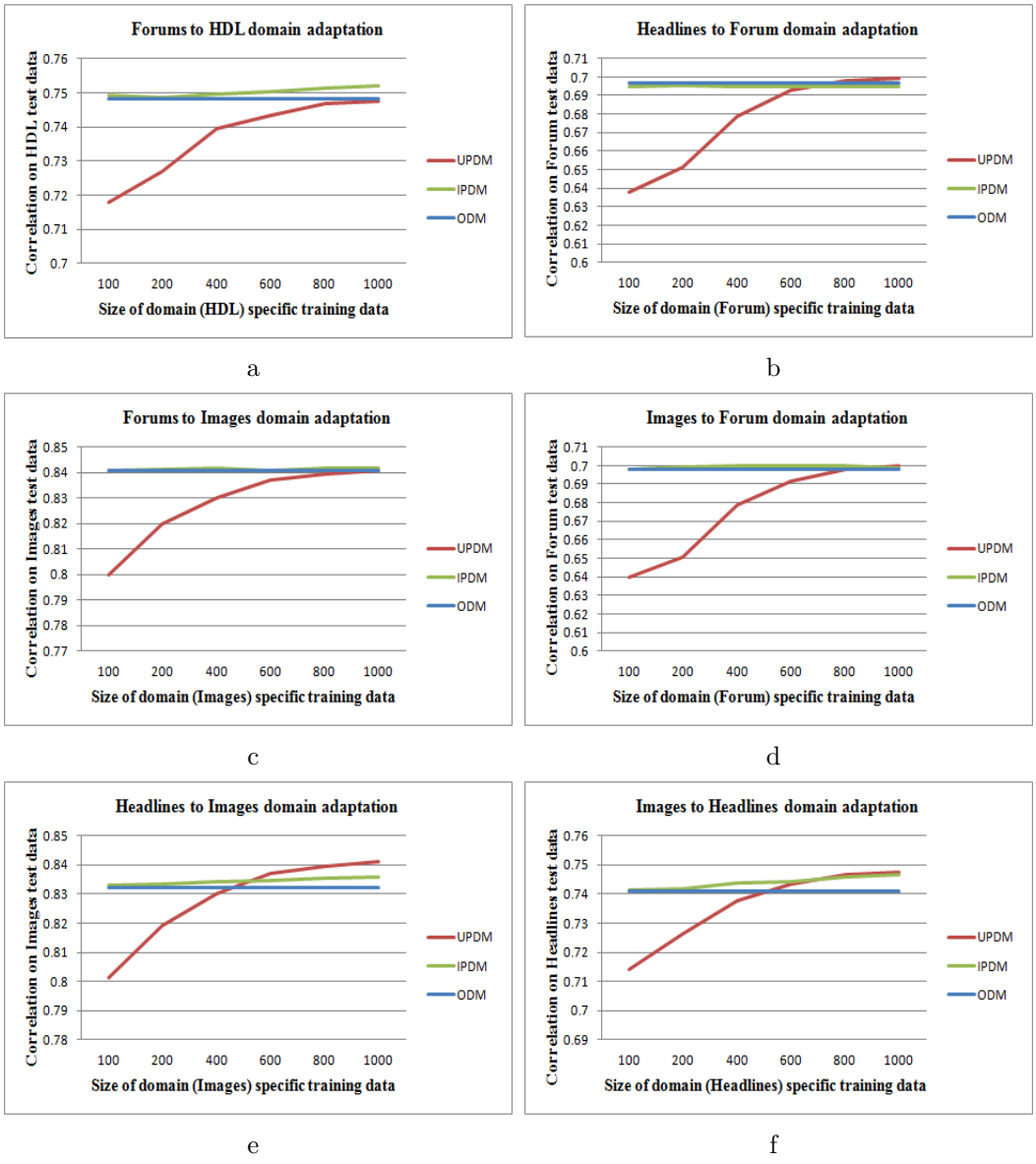


Fig. 2.8: Graphs showing the results of domain adaptation with varying size of domain specific training data for six different transformations corresponding to each permutation of Headlines, Forums, and Images data. The ODM results are invariant of domain specific training data but are displayed for the ease of comparisons with UPDM and IPDM.

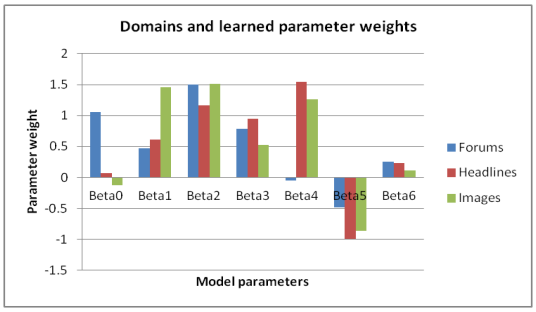


Fig. 2.9: The types of datasets (domains) and parameter weights learned using linear Bayesian models.

towards the right end, the results of IPDM are better than the ODM but are less than UPDM. But it still indicates informed prior domain model is better than out of domain model.

On the other hand, the $FRM \rightarrow IMG$ (in Figure 2.8c) results are more or less same when the domain specific training size is large. It indicates that the two domains are quite similar in nature. But it should be noted that there is clear (and consistent) win of using transfer learning approach particularly when the domain specific training data has up to several hundred records. It shows that the level of performance of ODM is good enough and is expected and the domain adaptation is useful in certain situations. But when compared UPDM and IPDM, the IPDM has performed better or at least same as UPDM in most of the cases, including the situation where the size of training data is large. This shows the advantages of transfer learning. But on the other hand, the frequentist methods do not allow us to use any prior knowledge and in the situation where the domain specific training data is small, the model will perform very poorly in domain specific test data.

We also built Bayesian linear models using training data of each of the domain and plotted the average values of samples of each model coefficients to get some insight into the difference in domains. They are presented in Figure 2.9. The model coefficients of Images data model and Headlines data look somewhat similar compared to Forum data. Though the density plots (in Figure 2.3) also show some

variations in distributions of scores across domains, in general, our datasets seem to have more similar pattern. However, we expect that the results will be different when the domains are quite different.

Overall, the results show the benefit of domain adaptation, most importantly, when the number of domain specific sample size is small and we are not sure about using out-of-domain data. We did not combine the out-of-domain data and the in-domain data for model building for two different reasons (a) the focus of this experiment was on using the transfer learning approach which is one of the salient features of Bayesian models, and (b) it is not always possible to retrieve the out-of-domain data but previously built (and published) model parameters maybe available as rough estimates and can be used with some confidence to initialize domain specific models and when the domain specific training data is large, the model parameters will get updated with no or little harm, if not beneficial, on the domain specific model.

2.7 Conclusion

In this chapter, we have presented our two different approaches to measuring the semantic similarity of texts at sentence-level. In one approach, we built Support Vector Regression (SVR) model with many features. Our models were evaluated by third person using a dataset that was not known during the model development. The correlation of our model predicted similarity scores and human judgments was up to 0.83 for some of the datasets in evaluation data released during SemEval 2016 STS challenge. Our system was one of the top performing systems among dozens of submitted systems in SemEval. Moreover, in comparison to many other systems, our models are more robust as we developed our models using a single set of data and applied to all the data in the evaluation set. In another words, we did not tune our models for each type of evaluation data.

In an another approach, we have modeled semantic similarity

probabilistically and proposed Bayesian models for measuring the similarity between sentences. We also proposed a domain adaptation approach using transfer learning. Our results indicate that the domain specific models adapting Gaussian mean priors learned from out of domain data performed better than the zero mean Gaussian models developed only from domain data as well as the domain general models, particularly when the domain data is small in size (< 600). Overall, the Bayesian models are more expressive than frequentist counterparts as they allow us to use the prior knowledge of the parameters which may be updated based on the observations and the similarity output is modeled probabilistically which is more interpretable compared to the point estimates.

Though we have not presented the results of our word-to-word similarity models, they are in fact integral part of many sentence similarity methods and we have worked on word level similarity as well. Furthermore, we have proposed other models and developed tools to facilitate measuring the semantic similarity. We will further discuss on these and our future directions in Chapter 7.

Chapter 3

Pooling Word Representations across Models

3.1 Introduction

Different approaches have been proposed over the years to represent words, phrases, sentences, or even larger texts in continuous vector forms (also called embeddings) (Landauer et al., 1998; Baroni & Zamparelli, 2010; Turian, Ratinov, & Bengio, 2010; Collobert et al., 2011; Mikolov, Chen, Corrado, & Dean, 2013; Pennington et al., 2014; Baroni, Dinu, & Kruszewski, 2014; Yu & Dredze, 2014; Iacobacci, Pilehvar, & Navigli, 2015). These vector based representations have been used in many Natural Language Processing (NLP) applications (Manning, Raghavan, Schütze, et al., 2008; Collobert et al., 2011; Socher et al., 2013; Lei, Xin, Zhang, Barzilay, & Jaakkola, 2014). Preferably, and which is often the case, the representations are derived in an unsupervised way from extremely large collections of texts. For instance, the pre-trained Word2vec (Mikolov, Chen, et al., 2013)[1] and GloVe (Pennington et al., 2014)[2] word vector representations were developed from texts containing billions of tokens covering millions of unique words: the pre-trained Word2vec model covers 3 million unique words, the GloVe model has a coverage of 1.9 million words, and a Latent Semantic Analysis (LSA) model developed ourselves from the whole set of Wikipedia articles (LSA_{wiki}; Stefanescu et al., 2014b)[3] contains representations for 1.1 million words[4].

While these are impressive numbers compared to manually created resources such as WordNet (Miller, 1995), it is important to note that the aforementioned word representation models share a limited number of words, as illustrated in

[1] https://code.google.com/p/word2vec/

[2] http://nlp.stanford.edu/data/glove.42B.300d.zip

[3] Wiki_NVAR_f7 at http://semanticsimilarity.org/

[4] We have used 'token' and 'word' interchangeably though they are not precisely the same.

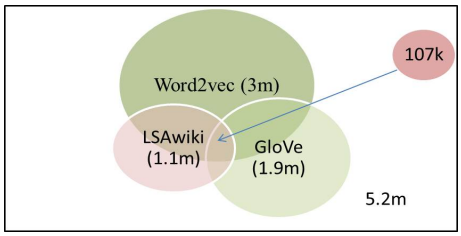


Fig. 3.1: Vocabulary coverage of three different pre-trained models (k - thousand, m - million).

Figure 3.1. The GloVe and Word2vec have about 154,000 words in common. Only about 107,000 words are common to all three models, which equates to only 3 to 10% of the words depending which model's vocabulary size is used as a reference. This clearly indicates that a significant chunk of words in each of these models are unique to the respective models and that they are missing from the other models. For example, the word "*Totalizator*" is present in Word2vec model but not in other two models. Therefore, systems using the LSA_{wiki} or GloVe model will have difficulty processing the word "*Totalizator*" because of the missing word representation whereas systems using Word2vec model will not encounter this situation, and so on.

Even though these numbers and the overlap in vocabulary among models can vary depending on the source of data used to build the models and the nature of preprocessing steps performed (e.g., lemmatization keeps only the base or dictionary form of the words), we will not have any single model that covers all the words in the web, for example. On the other hand, by design, many (if not all) existing NLP algorithms cannot work with multiple types of representations side by side. Accepting set of heterogeneous representations can greatly increase the complexity of such algorithms.

Yet another, better approach to the problem of missing word representations in vector based models, which we propose and explore in this chapter, is to automatically map word vector representations from one model (where they are

present; the *source* model) to another (where they are missing; the *target* model). We used Neural Network (NN) models for such mappings. That is, we make use of existing word representation models in combination with the NN-based mapping approach to extend the coverage (i.e., expand the vocabulary) of a given target model. The benefit of our approach is that we extend the coverage of a target model without the need to collect any extra texts and re-train the model, which could be non-trivial, as already mentioned, because such representation models are generally developed by different groups or organizations using non comparable set of corpora and obtaining all such corpora is not always possible due to various reasons including copyright and privacy issues.

Using our approach we can expand, for instance, the Word2vec, GloVe, and LSA models coverage to about 5.2 million unique words while showing that the transformed representations are well correlated (average correlation up to 0.801 for words in Simlex-999 dataset) with the native target model representations indicating that the transformed vectors can effectively be used as substitutes for native word representations of the target model. Also, the process can be automated if pre-trained word representation models are available. We rely on a novel Neural Network (NN) based approach to obtain vector-based representations for missing words in a *target* model from another model, called the *source* model, where representations for these words are available.

We have evaluated our approach *intrinsically* and *extrinsically* (see Section 3.4) on all possible *source* $\rightarrow$ *target* model permutations of three different pre-trained word vector models: word2vec, GloVe, and LSA_{wiki}. The results show that obtaining word representations for one model from another without much loss of representation power relative to the native target vectors is mostly possible and indicate that the transformed vectors can be used to augment the target models.

3.2 Related Work

The issue of handling unknown and missing words has been previously explored to some extent. Bengio, Ducharme, Vincent, and Janvin (2003) and Alexandrescu and Kirchhoff (2006) proposed deriving continuous word representations for unknown or missing words in Neural Language Models (NLMs) based on the words in context. However, (full) context of a word is not always available. Mikolov, Sutskever, et al. (2013) demonstrated that Word2vec vectors capture enough syntactic and semantic linguistic regularities to derive vector representations of missing words based on simple vector operations. For example, the following expression illustrates a singular/plural relation: *v('cats') = v('dogs') - v('dog') + v('cat')*. However, such nice linguistic regularities might not hold for complex and rare words and their vector representations might not be properly estimated (Luong, Socher, & Manning, 2013). Furthermore, it's hard to automatically find out such relations, such as in the case of proper nouns. Also, it will not work if certain word representations that are needed on the right hand side of an expression like the one above are not available.

Recursive Neural Networks (RNNs) have also been used to construct missing word representations from the vectors of words' morphemes (Luong et al., 2013). This approach works only if the missing word can be broken into morphemes, which in the case of some words such as proper nouns this is not possible, and representations for morphemes are available. In some of our models, we used representation of a word's synonyms, if exist, obtained from WordNet as a substitute representation for the target word (Banjade, Niraula, et al., 2015). However, this only works if representations for the word's synonyms exist, which is not always the case.

Though, to some extent, these techniques can handle the issue of missing word representations, the processes are not very straightforward to automate. Also, the increase in coverage will be limited as these methods have difficulty handling

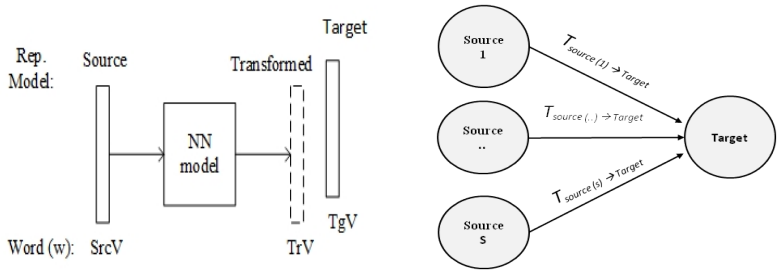


Fig. 3.2: Schematic diagram of (a) A transformation model, and (b) Multiple source-to-target transformations (NN- Neural Network, T - Transformation function/model, $SrcV$ - Source model vector, TrV - Transformed vector, TgV - Target model vector).

named entities which constitutes a large chunk of the vocabulary derived from very large corpora. In our mapping based approach, we directly transform word representations from one model to another model effectively pooling together their vocabularies (i.e., expanding each model's vocabulary). Additionally, this approach has potential to be equally applicable to phrase level or sentence level representations which have much more acute missing representation issues as the are even sparser than in the case of words.

3.3 Mapping Approach

As discussed previously and illustrated in Figure 3.1, words missing from a model may be present in another model. Therefore, by learning a word vector mapping model (or function) that can map one vector representation onto another, representations for missing words in the target model can be obtained from source model where the words are present. The schematic diagrams in Figure 3.2 illustrate this approach.

$$TrV = T_{source \rightarrow target}(SrcV) \tag{3.1}$$

$$Voc^{*}_{target} = Voc_{target} \cup (\cup_{i=1}^{S} Voc_{source(i)}) \tag{3.2}$$

In Eq. 3.1, $T_{source \rightarrow target}$ is a transformation model (function) corresponding to the $source \rightarrow target$ transformation. The transformation model consists of a

feed-forward Neural Network. The input to the model is in the form of source model vectors ($SrcV$) and the output of the transformation model (TrV) is similar to target model vectors (TgV). That is, ideally, the TrV should be the mirror image of TgV. The source vectors and target vectors can be of different types. For instance, the $SrcV$ can be LSA vectors while the TgV can be word2vec vectors and vice versa. Also, the dimensionality of the source and target vectors may be different.

And using S different source models (as depicted in Figure 3.2b), the effective size of the target model (Voc^*_{target}) will be increased greatly, particularly when there is less overlap among model vocabularies. For example, if one model is developed using corpus containing academic texts, such as Touchstone Applied Science Associates (TASA) corpus (Landauer et al., 1998) and another model is built from Wikipedia articles, then many words in Wikipedia will be missing in the TASA-based model (the vocabulary of Wikipedia-based model is a much larger than that of the TASA-based model). The proposed transformation model that can map vectors derived using Wikipedia onto the TASA-based model, thus greatly increasing the coverage of the latter. Similarly, the TASA-based model's word coverage can be further increased by adding other source models.

Specifically, we developed Neural Network models to map between any two of the following vector-based word representation models: LSA, word2Vec, and GloVe. There are six different transformations such as LSA-to-GloVe or word2vec-to-GloVe. It is important to note that these models are quite different in their underlying principles to derive word representations and that they are all unsupervised. LSA is an algebraic method. Word2vec is a feed-forward neural network based language model and its bag of word model utilizes the context of four words (two before and two after). GloVe model is based on algebraic as well as probabilistic approach theories.

The process of building mapping models and then mapping vectors from one

model to another can be automated allowing us to seamlessly use several word representation models as source to expand the vocabulary of a model of interest (see Figure 3.2b).

3.4 Evaluation Methods

We evaluated our transformation approach *intrinsically* and *extrinsically*. We used a simulation based approach in both cases, i.e., we simulated a set of missing words from an existing target model by removing them from it. This enables us to accurately assess the transformed vectors (TrVs) with respect to the native vectors in target model (TgVs). It should be noted that in the ideal case, the TrV would be same as TgV. That is, for developing and evaluating purposes of our NN based transformation models, we removed certain words that are common in the source (SrVs) and target model (TgVs) from the target model and then obtained representations for these words from the source model using our transformation model. We then compared the obtained transformed representations with the native representations from the target model (the vectors that were purposely removed) to check if they are alike (intrinsic evaluation) and perform similarly in NLP applications such as word-to-word similarity computations (extrinsic evaluation).

Intrinsic evaluation. We chose as our simulated missing words a set of N words that are present in both the source and target models so that the TrV could be directly compared to the TgV, i.e., the representations of the underlying target model itself. Then, we calculated an average correlation (r) score ($AvgCorr$) between the two vectors as shown in Eq. 3.3.

$$AvgCorr = \frac{1}{N}\sum_{i=1}^{N} r(TrV_i, TgV_i) \tag{3.3}$$

Extrinsic evaluation. For an extrinsic evaluation of the transformed vectors, we used a word-to-word similarity task which is one of the approaches used to measure the quality of word representations. If word representations are good, then similar

words will lead to high similarity score whereas dissimilar words will lead to low similarity score. Using a benchmark dataset containing pairs of words together with human-expert judgments of similarity and which is described in Section 3.5, we computed similarities between vector-based word representations ($Sim(V_{w1}, V_{w2})$) of words using the standard cosine similarity measure (normalized dot-product) applied on the transformed vectors TrVs of those words. Then, an overall correlation (r) between the similarity scores and human judgments' scores were computed as shown below.

$$TrSim = r(\{(Sim(TrV_{i1}, TrV_{i2}), H_i)\}) \tag{3.4}$$

$$TgSim = r(\{(Sim(TgV_{i1}, TgV_{i2}), H_i)\}) \tag{3.5}$$

where $1 \leq i \leq K$, K: size (# word pairs) of word similarity evaluation dataset, H_i: human rated similarity score for i^{th} word pair in the set.

We repeated the process using TgVs. When using the transformed vectors we obtained an overall correlation similarity score denoted as $TrSim$ and when using the native vectors the overall correlation score across all word pairs in our benchmark dataset is denoted as $TgSim$. A comparable $TrSim$ score to the $TgSim$ score would indicate that the transformed vectors can act as a substitute for word representations of the target model.

Baselines. We also used two baseline approaches to obtain transformed representations. A baseline approach used randomly chosen word vectors from the source model to transform onto the target model (we denote this transformation as $RandV@Src$). A second baseline approach used randomly chosen word vector representations from the target model itself without using any transformation model ($RandV@Trg$). The $RandV@Src$ and $RandV@Trg$ vectors were then compared with the actual, native word vector representations. These baselines help detect

whether the system is actually learning something or there is simply a random mapping.

3.5 Data

In this section, we start by briefly describing the word representation models which we chose for experimental evaluation of our proposed approach. Then, we describe the word-to-word similarity benchmark dataset and the training, evaluation, and test data generated from the word representation models for building and evaluating our transformation models.

Selected word representation models. We selected three different word representation models: (a) LSA model built using whole Wikipedia articles, (b) Word2vec model, and (c) GloVe model. These models were developed independently by different groups and were downloaded "as-is" without any intervention on our part as our purpose to take advantage of existing models without altering them in any way.

- **Word2vec**: This model is a pre-trained vector model based on the Google News dataset (about 100 billion words) and was developed by (Mikolov, Sutskever, et al., 2013) at Google. The distributed word vectors were computed using feed forward neural network based on a skip-gram model.

- **GloVe**: The GloVe (Global Vector), developed at Stanford University, is an unsupervised learning model for representing words (Pennington et al., 2014). The model was trained on non-zero elements in a global word co-occurrence matrix. We used the pre-trained model GloVe-42B which was trained on 42 billion words of Common Crawl corpus and it contains about 1.9 million unique tokens.

- **LSA**$_{wiki}$: We used the LSA model developed ourselves (Stefanescu et al.,

2014b) from the whole set of English Wikipedia articles (an early-2013 snapshot). The model was generated considering the lemmas of the content words that appeared at least 7 times in the corpus. This model contains 1.1 million unique entries.

All these models have 300-dimensional vectors, which, in the context of our research, is a pure coincidence as the dimensionality of various source and target models can be different.

Simlex-999. Simlex-999 (*Simlex*; Hill et al. (2014)) is a recently released dataset for word-to-word similarity evaluation. In this dataset, the related but semantically less equivalent word pairs are rated with low similarity scores by human judges. For instance, *lemon* and *tea* are related but not similar, and therefore, they are rated with low similarity score. The dataset consists of 999 word pairs. But some of the words in Simlex-999 dataset were not available in LSA_{wiki} and for consistency of our evaluation, we used only 955 word pairs that are available in all three word representation models. We used this dataset for both extrinsic and intrinsic evaluations.

Training, validation, and test datasets. From the pre-trained Word2vec, GloVe, and LSA models, we extracted 107,813 vectors corresponding to the common words in all three models (only 107,813 words were common in all three models). For each pair of models, we set-aside 1,017 *Simlex* word vectors for intrinsic evaluation and the remaining ones were randomly assigned for training (95,000 pairs of vectors), validation (5,000 pairs), and intrinsic evaluation (5,000 pairs or *5k-test*). *Simlex* words were used for both intrinsic as well as extrinsic evaluation. That is, we used two datasets (*Simlex*, and *5k-test*) for intrinsic evaluation. The difference in *Simlex* and *5k-test* is that the words in *Simlex* are curated words and contains only

Table 3.1: Summary of training, validation, and test datasets. Pair of vectors correspond to the words common to both source and target model. The information in this table applies to each transformation model.

Dataset	# pairs	Remarks
Training vectors	95,000	Used to build transformation model.
Validation vectors	5,000	Used for validating transformation model.
Simlex word vectors	1,017	Intrinsic evaluation (set 1).
5k-test vectors	5,000	Intrinsic evaluation (set 2).
Simlex words	955	Extrinsic (word-to-word similarity) evaluation.
Baseline ($Rand@Src$)	6,017	Source model vectors were randomly selected.
Baseline ($Rand@Trg$)	6,017	Randomly selected target model vectors used as TrVs.

common and meaningful words whereas the words in *5k-test* are randomly selected from the vocabulary containing millions of words. All the words in *5k-test* are not necessarily the meaningful ones (due to typos and other reasons) but this test set is bigger and practically more general. The remaining 1,796 vectors from the common vocabulary along with other (excluding vectors in training, validation, and test: *Simlex* and *5k-test*) randomly selected vectors (6,017 in total) from the corresponding source/target model were used for developing the baseline transformations, $RandV@Src$ and $RandV@Trg$. The vectors were normalized by their L^2-norms bringing them into the same scale. These datasets are summarized in Table 3.1.

3.6 Experiments and Results

We built NN models with a number of input units and output units equal to the size of the vectors in corresponding source and target models, respectively. They all were 300-dimension vectors (which was a pure coincidence and not a constraint of our mapping model). Therefore, the number of input units and output units were 300.

We added only one hidden layer keeping the model relatively simple and considering potential sparseness during training and performed experiments with varying number of hidden units. We developed those models using the neural network toolbox in Matlab (R2015a). The NN learning algorithm was set to the

Table 3.2: Results of vector transformation models (↓ - same as next rows, Std - Standard deviation).

Source→Target	**Word Similarity**		**AvgCorr (TrV, TgV) (Std)**	
	TgSim	TrSim	Simlex	5k-test
RandV@Src	↓	∼0.0	0.0-0.251	0.0-0.187
RandV@Trg		∼0.0	0.005-0.136	0.012-0.100
Word2vec→GloVe	0.427	0.446	0.748 (0.085)	0.488 (0.180)
LSA_{wiki}→GloVe		0.284	0.677 (0.092)	0.380 (0.160)
Word2vec→LSA_{wiki}	0.276	0.301	0.791 (0.104)	0.553 (0.214)
GloVe→LSA_{wiki}		0.292	**0.801** (0.103)	0.541 (0.217)
LSA_{wiki}→Word2vec	0.469	0.262	0.538 (0.089)	0.515 (0.147)
GloVe→Word2vec		0.369	0.676 (0.073)	**0.610** (0.116)

Scaled Conjugate Gradient (Møller, 1993) with logistic activation function and the number of iterations was set to 1,000.

Each *source* → *target* transformation model was trained using the training dataset of 95k pairs of vectors. We did experiments with different number of hidden units from 100 to 800 incrementing by 100. The $AvgCorr$ (see Eq. 3.3) on the validation set was used to calibrate the number of hidden units in the NN models. The results were improving with the increasing number of hidden units up to 600. However, the differences among the results with 400-600 hidden units were very small in all models[5]. Therefore, we chose to use 600 hidden unit models for all pairs of *source* → *target* models. We then evaluated the learned models on the test data (*Simlex* and *5k-test*). The results are summarized in Table 3.2. The $TgSim$ column presents the correlations (r) between the word similarities computed using target vectors and the human annotated similarity scores (see Eq. 3.5), for the word pairs in the *Simlex* dataset. The $TrSim$ column shows the same correlations for word similarities but this time using transformed vectors (see Eq. 3.4). The $TrSim$ scores when compared with $TgSim$ scores indicate how well the transformed vectors can act as a substitute for word representations in the target model. It is customary to

[5]In order to reduce the complexity of the model (or risk of overfitting), the number of hidden units could be set to 500 or 400 with small reduction in performance.

interpret the word similarity results in $TrSim$ with respect to $TgSim$ as the goal here is to have the transformed vectors that perform as good as the native target model vectors. In fact, *Simlex* is considered as a difficult dataset when compared with other popular word similarity evaluation datasets such as WordSim-353 (Finkelstein et al., 2001) because the related but not similar word pairs (e.g., *bread* and *butter*) in it are also assigned low similarity scores. And the correlation between similarity scores obtained using state-of-the-art word representation model and human judgment scores was found to be less than 0.5 for $Simlex$ (Hill et al., 2014).

We can see in Table 3.2 that the word-to-word similarity results using the transformed vectors ($TrSim$) are comparable or better in some cases with the results obtained using the native target model vectors ($TgSim$). For instance, the correlation between the similarity scores obtained using the native GloVe vectors and human judgments is 0.427 while the correlation (with human judgments) of similarity scores obtained using vectors transformed to GloVe from the Word2vec model is better at 0.446 (see Word2vec→GloVe in Table 3.2). However, some others, particularly the results obtained using transformations from LSA_{wiki} are relatively lower than those obtained by using the native target model vectors directly. For instance, $TgSim$ for GloVe is 0.427 but the results using the vectors transformed from LSA_{wiki} to Glove is 0.284 (in LSA_{wiki}→Glove). But still, the correlation of 0.284 can be considered as good given the difficulty of $Simlex$ data. Additionally, for each transformation we calculated correlation score between word-to-word similarity scores calculated for the $Simlex$ dataset using the target model vectors and the similarity scores calculated using the transformed vectors. This correlation score was up to 0.842 and it was for Word2vec→GloVe transformation. It was greater than 0.71 for four of the six different transformations and the lowest one was 0.633 for LSA_{wiki}→Word2vec. This indicates that the transformed vectors mostly

behave similar to the target model vectors in calculating word-to-word similarity and, therefore, the transformed vectors can be used to augment the target model.

Moreover, the average correlation score of TrVs with corresponding TgVs (in *AvgCorr* column) for *Simlex* words was up to 0.801 (in Glove→LSA$_{wiki}$) and it was up to 0.610 (in Glove→Word2vec) for *5k-test*. These correlation scores indicate that the transformed vectors closely resemble the target model vectors. In average, the *AvgCorr* scores across all transformations were 0.705 and 0.514 for *Simlex* words and words in *5k-test*, respectively. However, some transformations, particularly from LSA$_{wiki}$ yielded relatively lower scores. This is on a par with the word-to-word similarity results. The loss can be partly attributed to the transformation process.

But at the same time, the results of transformation from Word2vec and GloVe to LSA$_{wiki}$ were better (see Word2vec→LSA$_{wiki}$ and Glove→LSA$_{wiki}$). Therefore, it seems that the Word2vec model and the GloVe model we used were more effective than LSA$_{wiki}$ for the task we chose for the evaluation of our models (i.e., word-to-word similarity task). Likewise, it might mean that Word2vec and GloVe models are more powerful representation models than LSA$_{wiki}$ and it is hard to obtain representations of their form from LSA$_{wiki}$. But this information may not be sufficient enough to draw conclusions about this point as the scientific community is still striving towards finding a common ground on what measures to use for evaluating the word representation models and which word representation models are more powerful than others (Batchkarov, Kober, Reffin, Weeds, & Weir, 2016) by organizing events, such as RepEval (Representation Evaluation) workshop (Nayak, Angeli, & Manning, 2016).

Also, the correlations are relatively stronger and less spread (i.e, Std values are low) for *Simlex* than *5k-test*. It seems that some of the words in *5k-test* are not quite common (or meaningful) as compared to the *Simlex* words and transformation of such words' representations were not very effective. For example, we have

Table 3.3: Examples of words in *5k-test* set for which the correlation between Word2vec model representations and the representations obtained from GloVe by using our transformation model (GloVe→Word2vec) were high (on left), and low (on right).

Word	**Corr.** (TrV, TgV)
whimsical	0.8428
fashionable	0.8421
thinkers	0.8413
kayaking	0.8411
likable	0.8403
estrategia	0.8400
nicer	0.8396
whiny	0.8391
lovely	0.8391
puedes	0.8374

Word	**Corr.** (TrV, TgV)
poins	0.2225
witrh	0.2184
killingly	0.2159
poppermost	0.2120
pacifically	0.2046
witih	0.1945
tasman	0.1887
biolabs	0.1830
superboard	0.1734
pitchside	0.1472

presented, in Table 3.3, examples of words in *5k-test* for which the correlation between Word2vec model representations (TgVs) and the representations obtained from GloVe (TrVs) by using our transformation model GloVe→Word2vec are high (on left) and low (on right). We can see that the words on the right are rare words or misspelled words. Similarly, the correlation between TgV and TrV for misspelled word "*whihc*" found in *5k-test* is less than 0.35 in all $source \rightarrow target$ transformations.

Results for the $RandV@Src$ and $RandV@Trg$ baselines are presented as a range because the results were similar for all six different transformations. The highest average correlation ($AvgCorr$) was 0.251 for the GloVe to LSA_{wiki} transformation of *Simlex* words. In all other cases, the correlations were below 0.2. The word similarity results ($TrSim$) were around zero. These mean that providing random vectors from the source model as input or using randomly selected words for missing words in the target model has no significant outcome. Additionally, we checked the direct correlation between native source and target vectors in each case

but it was approximately zero when tested on *5k-test*. These indicate that learning a mapping function is needed.

3.7 Conclusion

We have presented a novel approach of expanding the vocabularies of word representation models by mapping vectors from one model (*source*) to another (*target*). Our results with three different pre-trained models indicate that the Neural Network based vector mapping approach is mostly effective. The average correlation between the target model's vectors and the transformed vectors was upto 0.801 for the words in Simlex-999 dataset. The extrinsic evaluation using word-to-word similarity task with Simlex dataset shows the results obtained using the transformed vectors are comparable with that of using the target model's representations. The results indicate that the transformed vectors mostly behave similar to the target model vectors and, therefore, the transformed vectors can be used with confidence to augment the target model.

Such type of mappings that vastly increases the coverage of a target model can be very useful in many NLP applications which most likely need to handle missing words or phrases. Our experiments with pre-trained models: Word2vec, GloVe, and LSA_{wiki} show that their representations can be pooled together to have vocabulary coverage of 5.2 million words for each model where the maximum number of words in a single model was about 3 million. However, this approach is particularly important when the representation models are developed from diverse corpora and there is less overlap in their vocabularies, and it can alleviate the problem of missing word representations in many NLP applications. Moreover, our approach is very straightforward and can be automated once a set of representation models are available.

Nevertheless, finding out whether certain type of source or target model makes transformations more or less effective is a topic of future investigation.

Additionally, the proposed solution can be used to obtain phrase representations which are even sparser than words. It is another interesting topic for future research.

Chapter 4

Open-Ended Answers Assessment in Tutorial Dialogue

4.1 Introduction

Open ended answers are responses produced by students to questions, e.g. in a test or in the middle of a tutorial dialogue. Such answers are very different from answers to multiple choice questions where students just choose one or more options from the given options and they are more easier to evaluate than open ended answers. In this chapter, we present Probabilistic Soft Logic (PSL; Kimmig et al., 2012) model for automatic assessment of open-ended answers in dialogue based intelligent tutoring systems. We also present an evaluation dataset (called DT-Grade; named after DeepTutor tutoring system) containing 900 responses that we annotated for their correctness (Banjade, Maharjan, Niraula, Gautam, et al., 2016).

Typically, automatic answer assessment systems assess student responses by measuring how much of the targeted concept is present in the student answer. To this end, subject matter experts create target (or reference) answers to questions that students will be prompted to answer and the semantic similarity between student's answer is measured with reference answer to evaluate the correctness of the answer. As discussed in Chapter 1, the true understanding of student answers is intractable as it requires collecting and inferring over a huge knowledge, including the linguistic knowledge, domain knowledge, and world knowledge. As a practical alternative, semantic similarity methods are applied. In this approach, the high similarity between student answer with reference answer indicates the answer is correct. Otherwise, the answer is partially correct, incorrect, and so on. This is in fact a de facto standard in automatic answer assessment (see Section 4.2.2). It is fast, does not require too much of information, and found to be effective in general.

However, the implied assumption in similarity based answer assessment is

that the student answer and the reference answer are self contained (i.e., grammatically and semantically complete). But almost always, the student responses depend on the context (at least broadly on the context of a particular domain) but it is more prominent in some situations. Particularly in conversational tutoring systems, the meanings of students' responses often depend on the dialogue context and problem/task description. For example, students frequently use pronouns, such as *they, he, she*, and *it*, in their response to tutor's questions or other prompts. In an analysis of tutorial conversation logs, Niraula et al. (2014) found that 68% of the pronouns used by students were referring to entities in the previous utterances or in the problem description. In addition to anaphora, complex coreferences are also employed by students. Also, in tutorial dialogues students react often with very short answers which are easily interpreted by human tutors as the dialogue context offers support to fill-in the blanks or untold parts. Such elliptical utterances are common in conversations even when the speakers are instructed to produce more syntactically and semantically complete utterances (Carbonell, 1983). By analyzing 900 student responses given to DeepTutor tutoring systems, we have found that about 25% of the answers require some contextual information to properly interpret them.

As illustrated in the Table 4.1, the student answers may vary greatly. For instance, answer $A1$ is elliptical. The *"bug"* in $A2$ is referring to the mosquito and *"they"* in $A3$ is referring to the amount of forces exerted to each other. Due to such variations in the answer, we augment the semantic similarity model by adding additional information, such as question difficulty. For instance, a high knowledge student answering many of the difficult questions correctly will probably answer the current question correctly.

We have proposed a Probabilistic Soft Logic (PSL) model, a version of Markov Logic Network (MLN; Richardson & Domingos, 2006), which works based

Table 4.1: A problem and some student answers to the given question. These examples were extracted from the records of student interactions with DeepTutor.

Problem description: A car windshield collides with a mosquito, squashing it.
Tutor question: How do the amounts of the force exerted on the windshield by the mosquito and the force exerted on the mosquito by the windshield compare?
Reference answer: The force exerted by the windshield on the mosquito and the force exerted by the mosquito on the windshield are an action-reaction pair.
Student answers:
A1. *Equal*
A2. *The force of the bug hitting the window is much less than the force that the window exerts on the bug*
A3. *they are equal and opposite in direction*
A4. *equal and opposite*

on logical and probabilistic reasoning framework. PSL allows us to model the complex interactions between the stochastic variables, such as student's knowledge level, question difficulty and the correctness of the student answer and perform inferencing over Probabilistic Graphical Model (PGM).

Also, the existing datasets (described in Section 4.2.1) contain pairs of student answers and reference answers annotated for their correctness and such student answers are mostly self contained. Therefore, in order to foster research in automatic answer assessment in dialogue context (also in general), we annotated 900 student responses gathered from an experiment with the DeepTutor intelligent tutoring system (Rus, Niraula, & Banjade, 2015; Rus, DMello, et al., 2013). We have made the dataset freely available for research purposes[1]. We also present a model which is based on word weighting and alignment based semantic similarity feature.

[1]http://language.memphis.edu/dt-grade

4.2 Related Work

In this section we discuss on the available datasets and the methods proposed in the literature.

4.2.1 DataSets

Nielsen, Ward, Martin, and Palmer (2008) described a representation for reference answers, breaking them into detailed facets and annotating their relationships to the learners answer at finer level. They annotated a corpus (called SCIENTSBANK corpus) containing student answers to assessment questions in 15 different science domains. Sukkarieh and Bolge (2010) introduced an ETS-built test suite towards establishing a benchmark. In the dataset, each target answer is divided into a set of main points (called content) and recommended rubric for assigning score points.

Mohler and Mihalcea (2009) published a collection of short student answers and grades for a course in Computer Science. Most recently, a Semantic Evaluation (SemEval) shared task called Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge (Dzikovska et al., 2013) was organized to promote and streamline research in this area. The corpus used in the shared task consists of two distinct subsets: BEETLE data, based on transcripts of students interacting with BEETLE II tutorial dialogue system (Dzikovska et al., 2010), and SCIENTSBANK data. Student answers, accompanied with their corresponding questions and reference answers are labeled using five different categories. Basu et al. (2013) created a dataset called Powergrading-1.0 which contains responses from hundreds of Mechanical Turk workers to each of 20 questions from the 100 questions published by the USCIS (United States Citizenship and Immigration Services) as preparation for the citizenship test.

Our work differs in several important ways from previous work. Our dataset is annotated paying special attention to dialogue context. In addition to the tutor question, we have provided the problem description as well which provides a greater

amount of contextual information and we have explicitly marked whether the contextual information was important to properly interpret/annotate the answer. Furthermore, we have annotated whether the student answer contains important extra information. This information is also very useful in building and evaluating natural language tools for automatic answer assessment.

4.2.2 Assessment Methods

Most of the earlier works on automatic answer assessment were focused on essay grading. Essays are typically open ended, i.e. expressing ideas or thoughts about some topic. On the other hand, short answers are mainly close ended, i.e. have a fixed target answer. We mainly discuss approaches for constructed answer assessment where the expected answer is short (one to just few lines) and reference answers are available to compare with in order to assess the student answer.

Martin and VanLehn (1995) proposed an assessment system OLAE using Bayesian nets. In OLAE, assessment produces a student model, i.e. a collection of correct and incorrect rules from the domain model known and used by a particular student. A student model is essentially a rule-based computer program that computes answers to actual problems in the same way as the student does. OLAE uses such an approach because assessments of which rules a student uses are necessarily uncertain. Though their solution is distinctive, the problem with this approach is that the human must generate the Bayesian network for each problem; this is why the approach does not scale.

The short answer grading has reached commercial levels as well. The C-rater system (Leacock & Chodorow, 2003; Sukkarieh & Blackmore, 2009) is one of the ETSs automatic scoring technologies (e-rater, c-rater, m-rater, and SpeechRater for essay scoring, content scoring, math scoring, and Speech input scoring respectively). C-rater is used for automatic analytic-based content scoring of short free-text responses. Analytic-based content is the kind of content that is predefined by a test

developer in terms of main ideas or concepts. These concepts form the evidence that a student needs to demonstrate as her/his knowledge in his/her response. C-Rater matches the syntactical features of a student response (subject, object, and verb) to that of a set of correct responses. Their system breaks the reference answers into constituent concepts that must individually be matched for the answer to be considered fully correct. The c-rater system has been used within many domains, including biology, English, mathematics, information technology literacy, business, psychology, and physics. The C-rater requires that the reference answer be broken down into a set of concepts in the form of simple sentences. Then, it applies textual entailment techniques based on syntax, lexical semantics, and simple semantic roles to identify whether the concept is present or not. However, the process is time consuming and requires more human effort. As they mentioned (Sukkarieh & Blackmore, 2009), the knowledge engineering process of building a model for a question took at least 12 hours. They proposed automatic model building for C-rater.

LSA (Landauer et al., 1998; Landauer, 2003) and machine translation evaluation methods are also applied for short answer grading. Pérez et al. (2005) applied the combination of Bleu-inspired algorithm and LSA. Their idea was to perform both syntactic and semantic analysis. They did some syntactic analysis such as stemming, closed-class word removal, word sense disambiguation and synonyms treatment procedures etc. They combined LSA method with syntax based methods where LSA captures the semantics. Despite the simplicity of these shallow NLP methods, they achieved significant correlations to the teachers scores while keeping language-independence and without requiring any domain specific knowledge. Another short answer grading system, used in AutoTutor system (Graesser et al., 2000) applied LSA approach. Later work on AutoTutor seeks to expand upon the original bag-of-words approach.

Mohler and Mihalcea (2009) explore unsupervised text similarity techniques for the task of automatic short answer grading answers to the introductory computer science assignments. They applied a number of knowledge-based (for example, WordNet) and corpus-based measures (LSA and ESA) of text similarity. They explored the impact of domain and size of the model development corpus on the accuracy. To evaluate the domain impact, they developed LSA model from Computer Science articles and compared with the LSA models developed from general Wikipedia articles. They found higher correlation of similarity score with human ratings when domain specific (i.e. model developed computer science articles) model was used. Mohler, Bunescu, and Mihalcea (2011) applied semantic similarity methods and dependency graph alignments to grade short answer questions. Similarly, Murrugarra, Lu, and Li (2013) proposed using domain general methods, bag-of-words approach, LSA representation, textual entailment, and others.

Rus and Lintean (2012) presented a novel, optimal semantic similarity approach based on word-to-word similarity metrics to solve the important task of assessing natural language student input in dialogue-based intelligent tutoring systems. The optimal matching is guaranteed using the sailor assignment problem, also known as the job assignment problem, a well-known combinatorial optimization problem. They compared the optimal matching method with a greedy method as well as with a baseline method on data sets from two intelligent tutoring systems, AutoTutor (Graesser et al., 2005) and iSTART (McNamara, Levinstein, & Boonthum, 2004). Creating a good set of reference answers is one point where the human involvement is needed in automatic answer grading. Student can express the same concept using various ways. The automatic answer grading would be done more confidently when the student answer is expressed similar to the reference answer(s). Utilizing the student answers to increase the pool of reference answer is a

possibility. Also, grouping the similar answers and evaluating them in a group requires less effort. Basu et al. (2013) have proposed a model on that called power-grading. They utilize the similarity methods used in answer grading to form clusters and sub-clusters. The answers in the same cluster are evaluated by teacher in a single action and similar feedback is given to the whole group.

As semantic similarity and textual entailment are closely related to the problem of automatic answer evaluation, virtually every text to text similarity and entailment method could be framed into this task. Various researches show that the similarity based methods can be potentially used in the answer grading tasks. In fact, a Semantic Evaluation (SemEval) shared task called Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge was organized in 2013 (Dzikovska et al., 2013) to promote and streamline research in this area and almost all of the participating teams applied semantic similarity techniques.

Although various results show that the similarity based methods can be used in the answer grading tasks, their implied assumptions are that the text available are standard texts with noise filtered. Our work is focused on using naturally occurring texts from conversational tutoring systems where various linguistic phenomena are present, such as coreferences (Raghunathan et al., 2010), ellipsis (Carbonell, 1983). We also augment the semantic similarity based model using additional knowledge.

4.3 Data Collection and Annotation

Data Collection

We created the DT-Grade dataset by extracting student answers from logged tutorial interactions between 40 junior level college students and the DeepTutor system (Rus, DMello, et al., 2013). During the interactions, each student solved 9 conceptual physics problems and the interactions were in the form of purely natural language dialogues, i.e., with no mathematical expressions and special symbols. Each problem contained multiple questions including gap-fill questions and short

constructed answer questions. As we focused on creating constructed answer assessment dataset with sentential input, we filtered out other types of questions and corresponding student answers. We selected 900 answers for the annotation. We chose the more difficult ones such that improving results on this dataset will greatly improve the assessment systems. Particularly, the similarity based models will have difficulty judging those answers.

Annotation

The annotation was conducted by a group of graduate students and researchers who were first trained before being asked to annotate the data. The annotators had access to an annotation manual for their reference.

Each annotation example (see Figure 4.1) contained the following information: (a) problem description (describes the scenario or context), (b) tutor question, (c) student answer in its natural form (i.e., without correcting spelling errors and grammatical errors), (d) list of reference answers for the question. The annotators were asked to read the problem and question to understand the context and to assess the correctness of the student answer with respect to reference answers. Each of the answers has been assigned one of the following labels.

1. **Correct**: Answer is fully correct in the context. Extra information, if any, in the answer is not contradicting with the answer.

2. **Correct-but-incomplete**: Whatever the student provided is correct but something is missing, i.e., it is not complete. If the answer contains some incorrect part also, the answer is treated as incorrect.

3. **Contradictory**: Answer is opposite or is very contrasting to the reference answer. For example, "equal", "less", and "greater" are contradictory to each other. However, Newton's first law and Newton's second law are not treated

<Instance ID ="386">
<MetaInfo StudentID = "DTSU017" TaskID = "LP03_PR09.bLK.sh"
DataSource = "DeepTutorSummer2014"/ >
<ProblemDescription>A car windshield collides with a mosquito, squashing it.
< /ProblemDescription>
<Question>How does Newton's third law apply to this situation?< /Question>
<Answer>both objects exert the same amount of force on each other.< /Answer>
<Annotation Label = "correct(0),correct_but_incomplete(0),contradictory(0),incorrect(0)"
<Additional Annotation ContextRequired = "0,1" ExtraInfoInAnswer = "0,1" / >
<Comments Watch = "0,1" > < /Comments>
< /Annotation>
<ReferenceAnswers>
The action is the windshield squashing the mosquito, and the equal and opposite reaction is the mosquito hitting the windshield.
< /ReferenceAnswers>
< /Instance>

Fig. 4.1: An annotation example where problem description, tutor's question, student's answer, and reference answer are shown.

as contradictory since there are many commonalities between these two laws despite their names.

4. **Incorrect**: Incorrect in general, i.e., none of the above three judgments is applicable. Contradictory answers can be included in the incorrect set if we want to find all kinds of incorrect answers.

Additionally, annotators were asked to mark:

- **ContextRequired**: whether contextual information was really important to fully understand a student answer. For instance, the student answer in the Figure 4.1 contains the phrase *"both forces"* which is referring to the force of windshield and the force of mosquito in problem description. Therefore, contextual information is useful to fully understand what both forces the student is referring to. As shown in Table 4.1 (in Section 4.1), a student answer could be an elliptical sentence (i.e., does not contain complete information on its own). In such cases, annotators were asked to judge the

student response based on the available contextual information and reference answers and nothing more; that is, they were explicitly told not to use their own science knowledge to fill-in the missing parts.

- **ExtraInfoInAnswer**: If a student response contained extra information (i.e., more information than in the reference/ideal answer provided by experts), we asked annotators to ignore the extra parts unless it expressed a misconception. However, we told annotator to indicate whether the student answer contains some additional important information such as a detailed explanation of their answer.

The annotators were encouraged to write comments and asked to set the 'watch' flag whenever they felt a particular student response was special/different. Such 'to watch' instances were considered for further discussions with the entire team to either improve the annotation guidelines or to gain more insights regarding the student assessment task.

The dataset was divided equally among 6 annotators who then annotated independently. In order to reach a good level of inter-annotator agreement in annotation, 30 examples were randomly picked from each annotation subset and reviewed by a supervisor, i.e., one of the creators of the annotation guidelines. The agreements (in terms of Cohen's kappa) in assigning correctness label, identifying whether the context was useful, and identifying whether the student answer contained extra information were 0.891, 0.78, and 0.82 respectively. In another words, there were significant agreements in all components of the annotation. The main disagreement was on how to use the contextual information. The disagreements were discussed among the annotators team and the annotations were revised in few cases.

Table 4.2: Summary of DT-Grade dataset. First part of the table shows the distribution of assessment labels and the second part shows the percentage of samples requiring context, and the percentage of answers having additional information than expected in reference answer.

Parameter	**Value**
All	900
Correct	365 (40.55%)
Correct but incomplete	209 (23.22%)
Contradictory	84 (9.33%)
Incorrect	242 (26.88%)
Requiring context	223 (24.77%)
Containing extra info	102 (11.33%)

The Dataset: We have annotated 900 answers. Table 4.2 offers summary statistics about the dataset. The 40.55% of total answers are correct whereas 59.45% are less than perfect. We can see that approximately 1 in every 4 answers required contextual information to properly evaluate them.

Next, we describe a base model where contextual word weighting approach is used in semantic similarity based assessment model. The PSL based model is described subsequently (in Section 4.5).

4.4 Contextual Word Weighting and Similarity Based Approach

Approach

Once the dataset was finalized we wanted to get a sense of its difficulty level. We developed a semantic similarity approach in order to assess the correctness of student answers. Specifically, we applied optimal word alignment based method (Banjade, Niraula, et al., 2015; Rus & Lintean, 2012) to calculate the similarity between student answer and the reference answer and then used that score to predict the correctness label using a classifier. In fact, the alignment based systems have been the top performing systems in semantic evaluation challenges on semantic textual similarity (Agirre et al., 2014, 2015; Han et al., 2013; Sultan et al., 2015).

The challenge is to address the linguistic phenomena such as ellipsis and

coreferences. An approach can be to use off-the-shelf tools, such as coreference resolution tool included in Stanford CoreNLP Toolkit (Manning et al., 2014). However, we believe that such NLP tools that are developed and evaluated in standard dataset potentially introduce errors in the NLP pipeline where the input texts, such as question answering data, are different from literary style or standard written texts.

As an alternative approach, we assigned a weight for each word based on the context: we gave a low weight to words in the student answer that were also found in the previous utterance, e.g. the tutoring systems question, and more weight to new content. This approach gives less weight to answers that simply repeat the content of the tutor's question and more weight to the answers that add the new, asked-for information; as a special case, the approach provides more weight to concise answers (see $A1$ and $A2$ in Table 4.1). The same word can have different weight based on the context. Also, it partially addresses the impact of coreferences in answer grading because the same answer with and without coreferences will be more likely to get comparable scores. The reference answers are usually self contained, i.e., without using coreferring expressions and only those student answers which are also self-contained and similar to reference answer will get higher score. On the other hand, answers using coreferences (such as: they, it) will get lower score unless they are resolved and the student answer becomes similar to reference answer. Giving lower weights to the words, if present in the student answer, for which student could use coreferrences makes these two types of answers somewhat equivalent.

Finally, the similarity score was calculated as:

$$sim(A, R) = 2 * \frac{\sum_{(a,r)\in OA} w_a * w_r * sim(a, r)}{\sum_{a\in A} w_a + \sum_{r\in R} w_r} \tag{4.1}$$

Where A/R refers to student/reference answer and a/r is a token in it. The $sim(a, r)$ referes to the similarity score between a and r calculated using word2vec

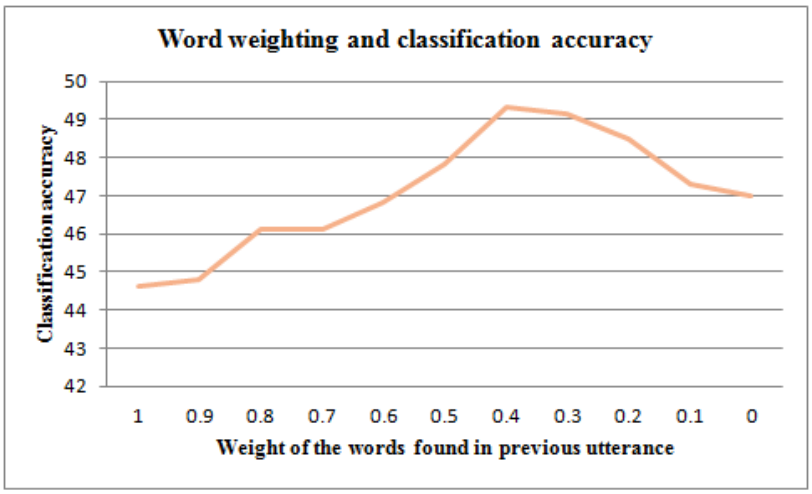


Fig. 4.2: Classification accuracy and weight of the words that are found in the last utterance.

model (Mikolov, Chen, et al., 2013). OA is optimal alignment of words between A and R obtained using Hungarian algorithm as described in Banjade, Niraula, et al. (2015). The $0 \leq w_a \leq 1$ and $0 \leq w_r \leq 1$ refer to weight of the word in A and R respectively.

Experiments and Results

In order to avoid noisy alignments, the word-to-word similarity score below 0.4 was set to 0.0 (empirically set). The $sim(A, R)$ was then used with Multinomial Logistic Regression (in Weka) to predict the correctness label. If there were more than one reference answers, we chose one with the highest similarity score with the student answer. We then set different weights (from 1.0 to 0.0) for the words found in tutor utterance (we considered a word was found in the previous utterance if its base form or the synonym found in WordNet 3.0 (Miller, 1995) matched with any of the words in the previous utterance). We changed the weight in the student answer as well as in the reference answer and the impact of weight change in the classification results were assessed using 10-fold cross validation approach. The changes in classification accuracy with changing weights are presented in Figure 4.2.

Giving weight of 1.0 to each word is equivalent to aligning words in student answer with the reference answer without looking at the context. But we can see the improvement in classification accuracy after reducing word weights up to 0.4 (accuracy 49.33%; kappa = 0.22) for the words found in the previous utterance and then decreases. It indicates that the words found in previous utterance should get some weight but new words should get more importance. This approach is somewhat intuitive. But deeper semantic understanding is required in order to improve the performance. For instance, sometimes this word weighting approach infers more information and gives higher weight to the incomplete utterance where student's true understanding of the context is hard to predict. Furthermore, it is non-trivial to use additional context, such as problem description including assumptions and graphical illustrations.

4.5 Probabilistic Soft Logic Model

Probabilistic Soft Logic (PSL) is an approach to combining knowledge in the form of first-order logic and Probabilistic Graphical Models (PGM) in a single representation. Probabilistic graphical models allow us to efficiently handle uncertainty and first-order logic allows us to compactly represent the knowledge. Furthermore, it allows to model the complex interactions among stochastic variables which is not possible to model in many other algorithms, such as Logistic Regression which treats each examples as i.i.d. (independent and identically distributed). But, for example, voting decision of friends has some influence on each other. Similarly, in answer assessment, a high knowledge student giving correct answers to the difficult questions will probably answer another difficult or easy question correctly and we model such knowledge in our PSL model. In specific, our model uses semantic similarity information augmented with other knowledge, such as question difficulty and knowledge level of the student.

First-Order Knowledge Base. A first-order knowledge base (KB) is a set of formulas in first order logic (Genesereth & Nilsson, 1987). Formulas are constructed using symbols: *constants*, *variables*, *predicates*, and $functions$. Constant represents an object (e.g., John). Functions represent mappings from tuples of objects to objects (e.g., $FatherOf$). Predicate represents relations among objects (e.g., $Friends$) or attributes of objects (e.g., $Smokes$). The formulas are typically written in clausal form (also known as conjunctive normal form (CNF)). For example,

$$Friends(x, y) \wedge Friends(y, z) \rightarrow Friends(x, z) \tag{4.2}$$

4.5.1 PSL Program

A PSL program consists of rules along with relative weights associated with them and the data (or observations).

$$5.0 : Friends(x, y) \wedge Friends(y, z) \rightarrow Friends(x, z) \tag{4.3}$$

$$2.0 : Friends(x, y) \wedge Colleague(y, z) \rightarrow Friends(x, z) \tag{4.4}$$

The rules are grounded using observations, i.e., each variable in the rules is assigned to all possible values in the observed data. For example, if there are three people: Joe, Bob, and Lili, then a grounded rule would look like,

$$5.0 : Friends(Joe, Bob) \wedge Friends(Bob, Lili) \rightarrow Friends(Joe, Lili) \tag{4.5}$$

Predicates in PSL porgram can have truth values in the range of [0 1], i.e., they are soft. For example, question difficulty can be defined in the range of 0 to 1. However, in Markov Logic Network (MLN) the predicates can have either true or false. That is, the constraints in MLN are harder than in PSL.

Furthermore, the prior knowledge can also be encoded as rules in the PSL

program. In our hypothetical example, let's assume that people who are neither friends of friends nor friends of colleagues can still be friends but the chances are very low. This can be coded in the PSL program as illustrated below. It should be noted that the weight to our prior is very low as our belief is that any two persons being friends to each other (given no additional information) is possible but less likely.

$$0.0001 : Friends(x, z) \tag{4.6}$$

The weights to the rules can be learned from the data itself. We discuss on this later. Next, we discuss some of the variables, predicates and rules we used in our PSL program.

Variables

s - Student id

a - Answer id (or just id) which uniquely identifies an instance in the dataset. It should be noted that the question belonging to a may be same as that of some other answer id b because the same set of problems were attempted by multiple students.

By convention, the variables are represented by lower case letters.

Predicates

- $SimHigh(a) \in 0/1$ - similarity of answer a with corresponding reference answer is high
- $SimMedium(a) \in 0/1$ -similarity of answer a with corresponding reference answer is medium
- $SimLow(a) \in 0/1$ - similarity of answer a with corresponding reference answer is low
- $PriorKHigh(s) \in 0/1$ - prior knowledge of the student s is high

- $PriorKMedium(s) \in 0/1$ - prior knowledge of the student s is medium
- $PriorKLow(s) \in 0/1$ - prior knowledge of the student s is low
- $QDifficultyHigh(a) \in [0\ 1]$ - question difficulty is high (fraction of students who answered the question corresponding to a incorrectly)
- $QDifficultyMedium(a) \in [0\ 1]$ - question difficulty is medium (fraction of students who answered the question corresponding to a correctly but incompletely)
- $QDifficultyLow(a) \in [0\ 1]$ - question difficulty is low (fraction of students who answered the question corresponding to a correctly)
- $AttemptedBySameStudent(a, b) \in 0/1$ - whether a and b were attempted by the same student
- $Correct(a) \in [0\ 1]$ - the truth value of answer a being correct
- $CorrectButIncomplete(a) \in [0\ 1]$ - the truth value of answer a being correct but incomplete
- $Incorrect(a) \in [0\ 1]$ - the truth value of answer a being incorrect

Some Rules and Priors

We present few rules with quite arbitrary weights. We learn the weights for those rules from the data which we present later in this section. The priors (starting with negation symbol $\sim$) specify the possibilities of being false. It should be noted that the weights do not have to sum up to 1.

$2.0 : SimilarityHigh(a) \wedge Qdifficulty Low(a) \rightarrow Correct(a)$

$0.1 : Correct(b) \wedge AttemptedBySameStudent(a, b) \rightarrow Correct(a)$

$3.0 : SimilarityLow(a) \rightarrow Incorrect(a)$

$0.004 :\sim Correct(a)$

$0.002 :\sim CorrectButIncomplete(a)$

$0.003 :\sim Incorrect(a)$

4.5.2 Data

We used DT-Grade dataset described in Table 4.2. Since the number of *Contradictory* answers was low in number, we collapsed them to more general category *Incorrect* and therefore, we have used just three different labels for the correctness: *Correct*, *CorrectButIncomplete*, and *Incorrect*. Also, we did not have pretest score for four students and we performed experiments with the rest 36 students' data which resulted 790 answers in the dataset.

4.5.3 Grounding

During grounding phase, all the variables in the rules are substituted with possible values from the observations (i.e., data). Figure 4.3 illustrates an example of a grounded graphical network for a student's data but the graph can grow very large. For instance, the nodes corresponding to correctness labels of each answer are actually 3 (*Correct*, *CorrectButIncomplete*, *Incorrect*) but in the graph they are represented by a single node CL. Similarly, the question difficulty QD has three values (high, medium, and low) and each one is actually represented by a separate node. Also, each student has attempted 20 questions in average (counting those in the DT-Grade dataset only) which makes the graph bigger than what is shown in the figure. We discuss on the scale of the network in Weight Learning section.

The shaded nodes in the graph are observed nodes which we call **evidence**,

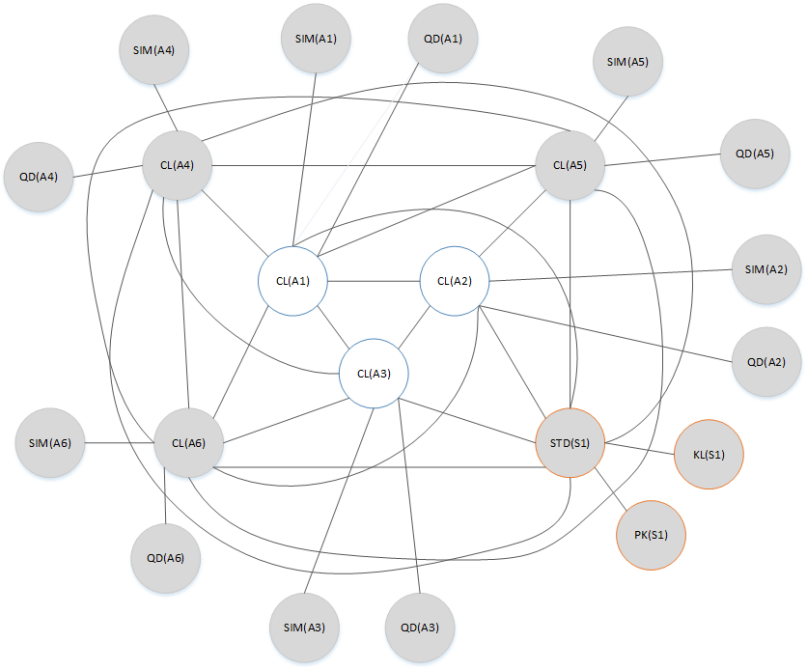


Fig. 4.3: An illustration of a grounded probabilistic graphical network for a student. The shaded nodes are *evidence* nodes and non-shaded nodes in the center are *query* nodes. CL - Correctness label, QD - Question difficulty, STD - Student, KL - (knowledge level), SIM - Similarity.

whereas the light nodes are **query** nodes. During inference, the truth values of the query nodes are predicted based on the evidence.

4.5.4 Weight Learning for PSL Rules

Laving out the internal details, the PSL rules' weight learning process is similar to typical supervised model learning process. We provide the ground truth (human annotated correctness label of the given answer) for the query predicate (which corresponds to a node in the grounded network graph). For each answer, there will be three query predicates one for each of the three labels. Since, the labels are mutually exclusive, only one of them is set to 1.0 while other two will be set to 0.0. As we discussed earlier, the grounded network can become very large depending on the set of rules and the size of the dataset used to ground the rules.

Also, the same node cannot be a query node as well as evidence node at the same time. Therefore, we have replicated each student's data several times and

renamed their ids such that we can make each answer in the original set (though renamed) a query at one time while using it as an evidence when other nodes are query nodes. This is important for us in both training (i.e., learning rules' weight learning) and evaluation phase because the dataset we used is comparatively small. For instance, if we make one answer for each student a query and make others as evidence, then we will have only 36 records for the weight learning as well as for the evaluation. But giving each node a chance to be a query node, we have the data several times bigger than the aforementioned size and we can also evaluate our model using full dataset (for example, by using leave-one-out approach). Actually, this process allows us to utilize the full set of data.

Just to get a sense of the scale of the graph, let's assume that each student's data is replicated 5 times. Then the size of the graph (by taking the dominant term only) will be $(5 * 790) * (5 * 790) \sim 1.5\ million$. Weight learning in such a huge probabilistic graphical model is impossible at least in our experimental settings. Therefore, we have pruned some rules and the resulting graph has about 200,000 nodes, on which we have managed to learn the weights for the rules.

For those rules which rapidly increase the size of the network with increasing size of the data, we have learned the weights for each student and estimated the weights of the rules using weights learned at student level which is the sub-optimal solution. For each student, the average size graph has about 35,000 nodes.

4.5.5 Experiments and Results

Including semantic similarity and additional information, we built several PSL models. We also performed experiments changing the priors learned separately. For the evaluation purpose, we took leave one student out approach.

The similarity between student answer and the corresponding reference answer was calculated using optimal word alignment based method which has performed very well in general (which we discussed in Chapter 2). The optimal

word alignment approach has been discussed in Section 4.4 also. We then grouped the similarity scores into high (score > 0.5), medium ($0.5 \geq$ score > 0.35), and low (≤ 0.35) using empirically chosen threshold values. Similarly, we have grouped the prior knowledge of the students into high (> 0.8), medium ($0.8 \geq$ score > 0.5), and low (≤ 0.5) based on their pretest scores. We also calculated the question difficulty (high, medium, and low) as discussed in Section 4.5.1. However, for question difficulty we have used soft values. In specific, each question has soft value for each of the difficulty levels: high, medium, and low. But for the difficult question, for example, the truth value of the predicate *QDifficultyHigh(a)* will have higher value than the truth value of the predicates corresponding to other difficulty levels (medium, and low).

By assuming the performance of a student is independent of others, we refactor the graph into subgraphs one for each student and take the leave one student out approach for PSL rules' weight learning and evaluation. As discussed in Section 4.5.4, we learned the weights for the rules from 35 students at a time (except for few rules for which weights were estimated using weights learned student-wise) and applied to the leave out student. Performing inference in such smaller graphs is computationally very efficient (takes few seconds for each student when run in a normal workstation). Also, it should be noted that the question difficulty was calculated based on training data only, i.e., using 35 students' data at a time.

Once inference is complete, i.e., the truth value for Correct, CorrectButIncomplete, and Incorrect predicates are assigned for each answer, we chose the correctness label corresponding to the highest truth value among those three. It should be noted that the truth value for each of them is in the interval [0 1] but their sum does not have to be 1.0. Then we calculated the accuracy and F1 scores. The results of our various models are presented in Figure 4.4.

The baseline system is the majority class classifier, i.e., which labels each

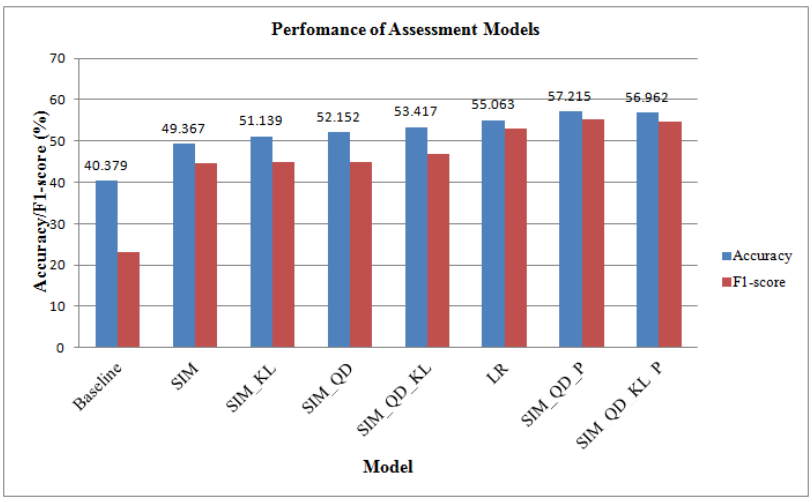


Fig. 4.4: Results of different Probabilistic Soft Logic models on DT-Grade dataset.

answer as correct. The accuracy of this baseline model was 40.379% which is equivalent to the percentage of correct answers in the dataset. SIM model used the similarity information only. It obtained 9% improvement over the baseline. As mentioned earlier, the DT-Grade dataset was developed by selecting the difficult cases, particularly difficult to judge by only comparing the student answer with the reference answer. Therefore, we consider 9% improvement in accuracy over baseline results as a notable improvement. We augmented the model using knowledge level (KL) of the student and question difficulty (QD). The KL includes the prior knowledge of the student which was assessed using multiple choice questions and the performance of the student on other than current question. While assessing correctness of an answer a given by student s, answer a is excluded while applying rules of types if a is correct, then b is also correct. Furthermore, the results were improved after adding question difficulty and knowledge level separately. When combined with similarity information, our model achieved 53.417% accuracy which is above 4% improvement over results obtained using similarity information only.

In an another experiment, we used the priors learned using Logistic Regression (LR). In specific, we obtained the probabilities of being Correct, CorrectButIncomplete, and Incorrect as predicted by LR model and used them in

our PSL models as priors (model names ending $\_P$). This has improved the results by about 5% in SIM_QD model and about 3% in SIM_QD_KL model. These results are also better when compared to the results of LR model. This shows that the LR model which is very different from PSL can complement the PSL model.

For our experiments, we used PSL tool [2] developed at University of Maryland, College Park. The tool uses hinge-loss Markov random fields (HL-MRFs) for inference and weight learning (S. Bach, Huang, London, & Getoor, 2013; S. H. Bach, Broecheler, Huang, & Getoor, 2015). We set the number of iterations to perform by the optimizer during rule learning to 50,000.

4.6 Conclusion

We have presented Probabilistic Soft Logic models for open-ended answer assessment. Our results show that PSL models built using semantic similarity information perform better than the baseline. We have also found that augmenting the PSL model with additional information, such as question difficulty and knowledge level improved the performance of assessment models by about 4%. Similarly, using the prior probabilities of being correct, correct but incomplete, and incorrect that are learned from logistic regression model further improved the results. Our models achieved accuracy up to 57% when evaluated with DT-Grade dataset which is unique and difficult dataset.

We also presented the dataset called DT-Grade which contains student answers given to the intelligent tutoring system and annotated for their correctness in context. We explicitly marked whether the contextual information was required to properly understand the student answer. We also annotated whether the answer contains extra information. That additional information can be correct or incorrect as there is no specific reference to compare with but the answer grading systems should be able to handle them. Additionally, we presented a system in which we

[2] http://psl.linqs.org/

used semantic similarity generated using optimal alignment with contextual word weighting as feature in the classifier for predicting the correctness label. The results of this model indicate that giving lesser weight to the words found in the recent utterances in the dialogue compared to the new content in the student answer improved the results.

Chapter 5

Negation Handling in Tutorial Dialogues

5.1 Introduction

According to SIL International (Summer Institute of Linguistics), negation is a morphosyntactic operation in which a lexical item denies or inverts the meaning of another lexical item or construction. A negator (or *negation cue*), is a lexical item that expresses negation. Morphological negation occurs when a word is negated by an affix (prefix or suffix) as in un-happy or sense-less whereas syntactic negation means an entire clause is negated explicitly (using a negator) or implicitly, e.g. verbs or nominalizations that negate their complements such as fail or deny. In explicitly negated statements, negation is marked using cue words, such as *not*, *no* and *neither ... nor*. A negation cue word or negator can affect the meaning of a part of the sentence in which it appears or part of previous sentence from the discourse context. The part of the sentence affected by the negation cue is called *negation scope*. The part of the scope that is most prominently negated is called *negation focus* (Huddleston et al., 2002).

An example of negation is shown in the following sentence where we indicate the negation cue (in <>), the negation scope (in []) and the negation focus (in {}): *The desk stops moving because [there is] <no> [net force acting on it].* Negation is a frequent and complex phenomenon in natural language. Tottie (1993) noted that negation is twice as frequent in spoken sentences (27.6 per 1,000 words) as in written text (12.8 per 1,000 words). Elkin et al. (2005) found that 12% of the concepts in 41 health records are negated while Councill, McDonald, and Velikovich (2010) report that 19% of the product review sentences contain negations. In an analysis of student utterances in dialogues collected from the Intelligent Tutoring System (ITS) DeepTutor (Rus, DMello, et al., 2013), it has been found that 9.36% of the student

answers contain explicit negation. The relative high frequency of negation and its key role in many applications such as intelligent tutoring, sentiment analysis, and information extraction emphasize the importance of the negation handling problem. In particular, the negation scope and focus can be used in semantic representations of negation, such as the one proposed by Blanco and Moldovan (2012).

Negation may become quite complex when interacting with other linguistic phenomena such as ellipsis and pragmatics, two frequent phenomena in dialogues, as illustrated in the example below. The example shows four different real answers ($A1$-4) as typed by high-school students during their interaction with the intelligent tutoring system DeepTutor.

Example 1

DeepTutor: Do these balls (red ball and blue ball) ever have the same speed?

A1: *They do not have the same speed.*

A2: *No.*

A3: *The balls never have the same speed.*

A4: *The red one goes faster.*

The four student answers are triggered by the same hint in the form of a question from the intelligent tutoring system. Answers $A1 - A3$ contain explicit negations whereas in answer $A4$ the negation is not explicit. We do not handle such cases, as in answer $A4$, as our focus is on explicit negation.

While datasets and computational approaches to negation have been recently developed, to the best of our knowledge, there is no previous work that systematically addresses the identification of negation scope and focus in dialogues. Previous work on computational approaches to negation have focused primarily on same-sentence negations, i.e. the scope and focused are in the same sentence where the negation cue word is (Morante & Daelemans, 2009; Morante, Schrauwen, &

Daelemans, 2011; Morante & Blanco, 2012; Thompson, Nawaz, McNaught, & Ananiadou, 2011; Vincze, Szarvas, Farkas, Móra, & Csirik, 2008). Our approach can detect negation scope and focus even when they reside in another sentence, i.e. the previous dialogue utterance. It should be noted that even when the scope and focus are in the same sentence as the negator, the context (of the dialogue in our case) could be helpful to correctly identify the focus.

We present in this chapter a method and negation dataset we created to handle negation scope and focus in tutorial dialogue (Banjade, Niraula, & Rus, 2016; Banjade & Rus, 2016). We collected and annotated a corpus from real dialogues between the computer tutor DeepTutor and high-school students. The corpus is called the DT-Neg corpus DeepTutor Negation corpus (Banjade & Rus, 2016)[1] - and consists of 1,088 instances. The corpus was manually annotated with negation cue words, negation scope, and negation focus. We then developed a method to detect negation scope and focus based on Conditional Random Fields (CRF; Banjade, Niraula, & Rus, 2016). We report results for focus detection with and without use of dialogue contextual features.

5.2 Negation in Dialogue

We argue that the scope and focus of negation in dialogue utterances is best determined in context. In this view, we adhere to the principle that the focus of negation is determined by coherence constraints in a discourse (Anand & Martell, 2012; ?, ?). That is, the scope and focus identification processes are informed by dialogue coherence constraints in the sense that, for instance, a word is preferred as a focus over another if it leads to better dialogue coherence. In our case, we use clues from previous dialogue utterances to help us disambiguate the scope and focus of a negation instance.

In Example 1, student answer $A1$ contains an explicit form of negation. The

[1]The dataset is freely available at http://language.memphis.edu/dt-neg

student answer is ambiguous in the sense that the focus switches from ever to have, given that ever is not mentioned by the student. That is, in one interpretation the student answer is understood as indicating that the two balls do not have the same speed (ever, i.e. ever is assumed to be implied by the student answer given the context of the tutor question). In another interpretation, the student answer $A1$ may be understood as indicating that the two balls do not have the same speed at some moment but may have the same speed at some other moment, which is the correct answer, by the way.

Answer $A2$ is a short answer. Such short answers are a typical case of ellipsis which is quite frequent in dialogue contexts, i.e. when words are elided from the student answer albeit implied by the context. Indeed, these types of negations in the presence of ellipsis can only be interpreted by considering the previous dialogue context which in this case is the tutor's previous question. Answer $A3$ is the cleanest form of negation because it is easiest to interpret as the student answer is self-contained and well-formed. $A4$ is an interesting answer in the sense that it does not contain an explicit negation. However, in the context of the previous question from the tutor this student answer is an indirect answer to the question. That is, in order to obtain the direct answer to the tutor question, answer $A4$ should be interpreted as "Because of the fact that the red one goes faster the two balls do not have the same speed, where we underlined the implied direct answer to the tutor question. This implied direct answer contains a negation. When analyzing negation in dialogues, the dialogue context will influence subtly the negation scope and focus. Consider the dialogue snapshot below.

- *Does the coin land in his hand?*

- *No.*

Because the focus of the question is asking where the coin will land, the

focus of the negation in the student answer is the location, i.e. hand. That is, the student is saying that the coin will land somewhere else (not in his hand).

Lets now consider the following dialogue snapshot:

- *Can you articulate the relevant principle?*

- *No.*

In this example, the computer tutor is specifically asking the student to articulate (not to apply) the relevant principle. Therefore, the focus is the verb articulate. One can also argue that the focus is the verb can. However, the clear intention of the "Can you articulate utterance from the intelligent tutoring system is an invitation to the student to articulate the principle, that is, the tutor's intention is actually "Please articulate the relevant principle. Since the invitation to articulate the principle maximizes the dialogue coherence, we choose articulate as the focus.

5.3 Related Work

Negation has been studied in the field of philosophy, psychology, linguistics, and computational linguistics starting with Aristotle (Wedin, 1990). Horn (1989) describes negation from philosophical and psychological perspectives, including constructs, usage, and cognitive processing of negation.

While logical negation has a very crisp definition (Horn, 1989; Rosenberg & Bergler, 2012), negation in natural language statements is more nuanced and subtle. Tottie (1993) presents a comprehensive taxonomy of clausal English negations denials, rejections, imperatives, questions, supports, and repetitions. Huddleston et al. (2002) have categorized the expression of negation into two types verbal or nonverbal, and analytic or syntactic in their book The Cambridge Grammar of the English Language. Miestamo (2006) distinguishes between standard negation and negation in imperatives, existential, and non-verbal clause.

Negation handling approaches were initially developed in the medical domain

for the purpose of processing and indexing clinical reports and discharge summaries. Mutalik, Deshpande, and Nadkarni (2001) developed Neg-finder in order to recognize negated patterns in medical texts. Chapman, Bridewell, Hanbury, Cooper, and Buchanan (2001) created a simple regular expression algorithm called NegEx that can detect phrases indicating negation and identify medical terms falling within the negative scope. Morante and Daelemans (2009) proposed a method of learning the scope of negation in biomedical text. Many other research works in negation handling focused on the medical domain (Gindl, Kaiser, & Miksch, 2008; Mac Namee, Kelleher, & Delany, 2008; Rokach, Romano, & Maimon, 2008). Vincze et al. (2008) annotated negation cues and their scopes in the BioScope corpus. The corpus consists of medical free texts, biological full papers and abstracts.

Negation was also studied in the context of sentiment analysis. Councill et al. (2010) focused on explicit negation and created a product review corpus annotated with negation cue and scope. Others have studied content negators, such as "hampered" and "denied" (Choi & Cardie, 2008; Moilanen & Pulman, 2007). Since identification of negation in review texts can help opinion mining tasks, Konstantinova et al. (2011) annotated the SFU Review Corpus.

In 2011, Morante, Schrauwen, and Daelemans published a more comprehensive set of guidelines for the annotation of negation cues and their scope. In fact, one of the shared tasks in the *SEM 2012 conference was dedicated to negation scope and focus detection (Morante & Blanco, 2012). Blanco and Moldovan (2012) annotated negation focus on text extracted from the PropBank corpus and the resulting dataset was used in the shared task (Morante & Blanco, 2012). Many of the participating teams adopted machine learning techniques for cue, scope, and focus detection. Some others used rule based systems as well. Although some of the evaluated approaches showed good performance on that

dataset, it is not clear whether those systems perform well in general as they were evaluated only with narrative, non-dialogue texts.

Zou, Zhou, and Zhu (2014) showed the importance of discourse context for negation focus detection but their work was limited to focus detection when the focus and negator are in the same sentence. But we approach the tasks of scope and focus detection for intra- and inter-sentential negation in dialogue.

5.4 Data Collection and Annotation

We created the DT-Neg dataset by extracting student answers containing explicit negation cues from logged tutorial interactions between high-school students and the DeepTutor tutoring system. During the interactions, students solved conceptual physics problems, as opposed to quantitative problems, and the interactions were in the form of pure natural language texts (i.e., no mathematical expressions and special symbols were involved). Each problem contained multiple questions. In 27,785 student responses, we found 2,603 (9.36%) student responses that contained at least one explicit negation cue word, such as *no* and *not*. We have not considered affixal negations, such as *un* in *un-identified*.

We tokenized the dialogue utterances using Stanford CoreNLP Toolkit (Manning et al., 2014). As we focused on explicit negation, we identified student answers containing negation cue words based on a list of cue words which we compiled from different research reports (Morante et al., 2011; Vincze et al., 2008) as well as our own data. If a student response contained multiple negations, they were treated as separate instances in our corpus. We then annotated each such candidate negation instance for negation cue, scope, and focus.

Annotation procedure. During annotation, annotators were asked to validate the automatically detected negation cue words and identify the corresponding negation scope and focus. It should be noted that we only targeted student responses for negation handling and not all the dialogue utterances, because

the system/tutor utterances are system generated and therefore their interpretation is known.

The annotation was conducted by a group of 5 people comprised of graduate students and researchers who were first trained before being asked to annotate the data. They had access to an annotation manual during actual annotation for reference. The guidelines have been inspired from the one prepared by Morante et al. (2011) for non-dialogue texts. We have developed our guidelines to best fit the context of our work, i.e. dialogues.

Annotators were instructed to use contextual information to best disambiguate the scope and focus. For this, annotators were shown the student response containing the negation as well as the previous system turn (tutor question). The Example 2 and Example 3 below illustrate annotations where in one case the negation scope and focus are in the same sentence as the negation cue word (Example 2) whereas in the other (Example 3) the negation scope and focus are located in the dialogue context, i.e. the previous dialogues utterance generated by the tutor. The cue, scope, and focus are marked by <>, [], and {}, respectively.

Example 2:

Question: Do these balls (red ball and blue ball) ever have the same speed?

A: *[They do] <not> [have the same speed].*

Example 3:

Question: Do [these balls (red ball and blue ball)] ever [have the same speed]?

A: *<No>*.

The annotators agreement for a scope location judgment, i.e. the same sentence or in the previous utterance was very high at 94.33%. When the annotators agreed on the location of scope and focus, we measured the agreement for scope and focus, respectively. The average token (sentence) level agreement was 89.43% (66.60%) and 94.20% (66.95%) for scope and focus, respectively. The main

Table 5.1: Summary of DT-Neg dataset.

Parameter	Training	Test
# instances (total)	761	327
#instances with scope/focus in context	328	130
# unique cues	10	9

disagreement among annotations was on how to use the contextual information. The disagreements were discussed among the annotators and fixed. The role of the discussion was to both reach an agreement and improve consistency of future annotations. In total, we have annotated 1,088 valid instances (an instance is a pair of tutor question and student answer).

We randomly divided the data into training and test set in 70-30%. General characteristics of the DT-Neg corpus are offered in Table 5.1. Different forms of the same cue, such as n't or not or NOT were considered identical while counting unique cues. We can observe that 42% of the instances in DT-Neg dataset have scope and focus in context (i.e., they are inter-sentential negations).

5.5 System Description

We have modeled negation scope and focus detection as a sequence labeling task in which each word in the negated sentence is either labeled as in-scope/focus or out-of-scope/focus. We used MALLET SimpleTagger (McCallum, 2002) which is a Java implementation of Conditional Random Fields (CRFs). CRF is a discriminative method for sequence labeling. It has been successfully applied in a number of sequence labeling tasks such as POS-tagging, and Chunking. It defines conditional probability distributions $P(Y|X)$ of label sequences Y given input sequences X. In our case, Y is a set of binary decisions about a token in the sentence where the negation scope/focus lies and X is the input sequence represented as a set of features. CRFs models may account for the full context of a set of observations such as the labels of tokens before and after the current token in the sentence. For instance, if a given token in a phrase is labeled as within the negation scope, the

probability of other tokens in the same phrase being in the negation scope will be high. Therefore, CRF is a best choice to label scope/focus when expert-labeled data are available to train the model.

Features. Each token in the student answer where the negation is present has a set of features which includes positional, lexical, syntactic, and semantic information. The following features were used for CRFs modeling and labeling purposes.

1. **Cue** − the negation cue itself (multiple words in the cue, such as neither nor, were merged together).

2. **Before cue** − whether the current token appears before the cue (first cue word if the cue has multiple words).

3. **Distance from the cue**− how far the current token is from the cue. Word next to the cue word has distance of 1.

4. **POS tag** − Part-of-speech tag of the token.

5. **Conjunction in between** − whether there is a conjunction (coordinating or subordinating) in between the token and the negation cue.

6. **Punctuation** − whether the token is punctuation.

7. **Student Answer type (1/0)** − short versus full sentence; this features suggest whether to look in the student answer for the scope and focus or in the previous utterance.

8. **Dep1** − whether there is a direct syntactic dependency between the current token and the cue word.

9. **Semantic role** − semantic role of the token based on head verb of the sentence.

10. **First word of question** − wh-word or first word of previous tutor utterance.

11. **Head word of question** − the lemma of the head word of the question obtained from the dependency parsing.

12. **Found in Question** − whether the word (stop-words are ignored) in its lemmatized form is found in question.

We will refer to these features by their numeric ids. Also, we categorize these features into the following groups: basic features (1-3), syntactic-semantic roles features (4-9, 9), and contextual features (10-12). We used Stanford CoreNLP Toolkit (Manning et al., 2014) to extract POS tags, dependency information, and head words. Semantic roles were generated using SENNA tool (Collobert et al., 2011).

Models and Evaluation

The training examples consist of tokens, associated features, and scope labels (using IOB2 format where the B- prefix before an in-scope/focus tag indicates that the tag is the beginning of the scope, and an I- prefix before a tag indicates that the tag is inside a scope/focus and O indicates that a token is outside of the scope/focus). Scope labels were removed from the test examples as the goal is to discover the labels automatically. As discussed earlier, the focus of negation may depend on the context even if it is in the same sentence where the negation cue word is (intra-sentential negation) or not (inter-sentential negation). The type of the previous question from the intelligent tutoring system (or another conversational partner in the general case of a dialogue system), the head word of the previous tutor question, and information about whether the word in the student answer is found in previous utterances are used as contextual clues in our model.

To measure the performance of the proposed models, we adopted the token label scoring used in *SEM 2012 Shared task (Morante & Blanco, 2012). We ignored punctuations when computing token label performance. A training-testing methodology was followed in which we first cross-validated the models using training data and then evaluated their performance on separate, previously unseen testing data. The default settings of CRF in MALLET (version 2.0.7) tool were used during model development.

Table 5.2: Results of negation scope detection system with DT-Neg dataset (SDR - Scope Detection Run).

System/Features	Precision	Recall	F1
Baseline	57.87	1.00	73.31
SDR1/1-3	76.97	95.89	85.40
SDR2/4-9	80.83	81.56	81.19
SDR3/1-9	90.80	95.31	93.00
SDR4/1-12	92.97	95.74	**94.34**
SDR5/1-3, 10-12	83.64	92.92	88.04

5.6 Experiments and Results

Scope detection (SD). Results (Precision, Recall, and F-measure) for scope detection are summarized in Table 5.2. In Run 1 (SDR1), we used just the basic features. In Run 2 (SDR2) syntactic and semantic role features were used. Runs SDR3 and SDR4 combine basic and syntactic-semantic role features with and without the contextual features. Run SDR5 uses basic features and contextual features. The baseline results were generated by labeling all tokens as they were in the negation scope.

As can be seen from the table, all of our systems performed significantly better than the baseline system. The combination of basic and syntactic-semantic features produced an F1 score of 93.00 and adding contextual features improved the results. The modest improvement when adding contextual features on top of the basic and syntactic-semantic role features could be due to the fact that we used a limited number of contextual features or it might be the case that the performance of the SDR3 model is already very good and significant improvement is difficult to obtain without an extremely rich model that would include many more contextual features or that the features have limited power. It could also mean that for scope detection syntax and semantic roles features play a more important role than our limited set of contextual features. To find a more precise answer to this latter hypothesis we analyzed the performance of a model (SDR5 in Table 6.2) that

Table 5.3: Results of focus detection system with DT-Neg dataset (S - scope used, FDR - Focus Detection Run).

System/Features	Precision	Recall	F1
Baseline	17.04	98.03	29.03
FDR1/1-9, S	77.06	75.54	76.29
FDR2/1-12, S	80.82	81.00	**80.91**
FDR1-Intra	76.60	81.52	78.98
FDR2-Intra	83.88	81.52	**82.68**
FDR1-Inter	80.00	51.67	62.80
FDR2-Inter	77.41	80.38	**78.87**

excluded the syntactic and semantic roles features. By comparing the performance of SDR1, SDR5, and SDR4 we can notice that adding the contextual features to the basic features model (SDR1) leads to an almost 3% improvement in the F1 measure. The further addition of the syntactic and semantic roles features to the SDR5 model that includes the basic and contextual features leads to a more than 6% improvement.

Focus detection (FD). The results for focus detection are summarized in Table 5.3. In this case, we used the same set of features (i.e., features 1-12). In addition, for focus detection we rely on scope labels obtained with the best performing scope detection model (i.e. SDR4 in Table 6.2) as we assume that the focus is within the scope. The baseline model was developed by treating all the in-scope tokens predicted by the best system (SDR4) as they were also in the focus.

Compared to the scope detection, the results suggest that focus detection is more challenging and it requires more context to best disambiguate it (we can see that by comparing FDR2 and FDR1 results). In an another experiment, we extracted from the DT-Neg corpus only instances in which the scope and focus were in the same sentence as the negation cue word, i.e. similar to how previous data sets treated negation. This allowed us to gauge the importance of context for same-sentence focus detection. Rows with the mark Intra denote this Answers-only focus subset, which includes 197 instances from the test set. By comparing results

of FDR1-Intra and FDR2-Intra, we can see that context can improve results and therefore is important for focus detection. Furthermore, we tested the role of contextual features on the remaining instances (i.e., instances where the negation focus itself lies in the context). These results are presented in the FDR1-Inter and FDR2-Inter rows. In this case also, contextual information improved the results.

5.7 Discussion and Conclusion

The proposed method for negation scope and focus detection in dialogue performed very well. Specifically, the results show that the contextual information in intra- and inter-sentential negation focus detection is important. This can be very useful towards improving natural language understanding in conversational (i.e., dialogue based) systems.

However, there are still issues that must be addressed. For instance, some of student responses were not well formed which introduce errors in our feature extraction step. Moreover, as the MITRE Corporation noted in their recent report, there are still some issues with respect to negation annotation and evaluation (Wu et al., 2013) that need to be addressed by future research. For example, previously existing datasets assumed negation scope is within the same sentence with the negation cue word (or at least annotated so) which does not generalize across all kind of data. We addressed this issue in our work presented here. Also, there may be inconsistencies in annotations proposed by various teams. For example, some negation corpora include cues within the scope whereas others don't. We did not include cue in the scope.

In order to foster research in this area, we intend to make our annotated dataset and annotation proposal freely available.

In the future, we want to work with datasets from different sources and work on the interpretation of negated texts in dialogue contexts which is an important task once negation scope and focus have been identified. For example, we plan to

handle negation in automatic answer grading systems in conversational tutoring system.

Chapter 6

Towards Interpretable Similarity and Diagnostic Feedback Generation

6.1 Introduction

Measuring the semantic similarity of texts is to quantify the degree of semantic similarity between a given pair of texts which we discussed in Chapter **??**. For example, a similarity score of 0 means that the texts are not similar at all and 5 means that they have same meaning, and so on. While useful, such quantitative or even qualitative assessments are hard to interpret because they do not provide details, i.e., they do not explain or justify why the similarity score was assigned high or low. If we look at the output of the automatic answer assessment systems, the final outcome is either the similarity score between student answer and the reference answer, or a label (such as Correct, CorrectButIncomplete, or Incorrect). But if the answer is not correct, the student may not be able to figure out what exactly was wrong in his or her answer. On the other hand, human tutor can generate explanation for the partially correct or incorrect answer. Towards achieving that goal of being able to generate the explanation for the given assessment score or label, we have developed interpretable similarity models and tools (Banjade, Niraula, et al., 2015; Banjade, Maharjan, Niraula, & Rus, 2016; Maharjan et al., 2016) which we describe in this chapter.

One way to provide an explanatory layer to text similarity assessment methods is to align chunks (phrases can be loosely called chunks) between texts and assigning semantic relation to each alignment. To this end, Brockett (2007) and Rus et al. (2012) produced datasets where corresponding words (or multiword expressions) were aligned and in the latter case their semantic relations were explicitly labeled. We align chunks, indicating the semantic relation (see Table 6.1) and the similarity score between chunks. The semantic relations were proposed as

Table 6.1: Types of semantic relations between chunks.

Semantic Relation	Description
EQUI	Chunks are semantically equivalent
SPE1/2	Chunk in sentence 1/2 is more specific than chunk in sentence 2/1
OPPO	Opposite in meaning
SIMI	Similar meanings, but not EQUI, OPPO, SPE
REL	Related meanings, but not SIMI, EQUI, OPPO, SPE
NOALI	Has no corresponding chunk in the other sentence

part of the SemEval challenge for interpretable similarity (Agirre et al., 2015, 2016) and are more comprehensive. Also, though less frequent, we allow one-to-many alignments.

For example, given the following two sentences (source: Agirre et al. (2016)),

12 killed in bus accident in Pakistan

10 killed in road accident in NW Pakistan

They are first chunked using some chunking tool (see Section **??**).

[12] [killed] [in bus accident] [in Pakistan]

[10] [killed] [in road accident] [in NW Pakistan]

Once chunks are obtained, we align the chunks and assign similarity score in the range of 0 to 5 and the output looks as shown below

[12] <=> [10] : (SIMI 4)

[killed] <=> [killed] : (EQUI 5)

[in bus accident] <=> [in road accident] : (SPE1 4)

[in Pakistan] <=> [in NW Pakistan] : (SPE2 4)

Our system (Banjade, Maharjan, Niraula, & Rus, 2016; Maharjan et al., 2016) which is also freely available for download [1] performed overall best in SemEval 2015 and 2016 (Agirre et al., 2015, 2016).

Using this approach, we target to produce detailed feedback as illustrated below.

Student answer: *The force on the box is zero*

[1]from http://semanticsimilarity.org

Table 6.2: The summary of training and evaluation dataset.

Dataset	Training	Test	Source
Images	750	375	image captions
Headlines	750	375	news headlines
Student Answers	333	344	student answers

Reference answer: *The net force on the box is zero*

Alignments:

The force <=> *The net force* (SPE2)

the box <=> *the box* (EQUI)

zero <=> *zero* (EQUI)

The **tutor feedback** would look something like below:

*Great! But to be **specific**, you should state "the net force"* (because the expected answer is more specific than student's answer, i.e., the $netforce$ and $force$ are quite different concepts in science and $netforce$ is more specific than $force$).

6.2 Dataset

We used the dataset released during SemEval competitions in 2015 and 2016. We used the dataset released during 2015 as training and evaluated the system using the dataset released in 2016. The dataset is summarized in Table 6.2. Each pair in the dataset is in plain text as well as chunked. These chunks were created manually (i.e., reference chunks or gold chunks). If the text is not chunked, system should be able to chunk them first before doing any alignments and the system generated chunks are referred as sys chunks. Our system can process the chunked text (*gold chunks category*) as well as in plain text which is first chunked by our system itself (*sys chunk category*).

6.3 Preprocessing

Hyphens were replaced with whitespaces if they were not composite words (e.g. video-gamed). Also, the words starting with co-, pre-, meta-, multi-, re-, pro-, al-, anti-, ex-, and non- were left intact. Then, the texts were tokenized, lemmatized,

POS-tagged and annotated with Named Entity (NE) tags using Stanford CoreNLP Toolkit (Manning et al., 2014). We also marked each word as whether it was a stop word. In the system chunks category, we had plain texts and we created chunks using our own Conditional Random Fields (CRF) based chunking tool (see Section 6.4). We normalized texts using mapping data. For example, *pct* and *%* were changed to *percent*. These preprocessing steps were performed for both gold chunks and system chunks category.

In student-answers dataset which consists of student answers given to a computer based logic tutor (Agirre et al., 2016), we replaced symbol *A/B/C* with *bulb A/B/C*. Similarly, *X/Y/Z* was replaced by *switch X/Y/Z*. We used this domain knowledge based on the notes found in student-answers training data description file.

6.4 Chunking

We developed a Conditional Random Field (CRF) based chunker[2] using both CoNLL-2000 shared task training and test data[3]. This data consists of a Wall Street Journal corpus: sections 15-18 as training data (211727 tokens) and section 20 as test data (47377 tokens). We generated shallow parsing features such as previous and next words from current word, current word itself, current word POS tag, previous and next word POS tags and their different combinations as described in Tjong Kim Sang and Buchholz (2000). We used CRF++ tool[4] to build the CRF models.

Furthermore, we analyzed its output (i.e., chunks) and added the following rules in the system to merge some of the chunks, resulting in chunks that make more sense and are consistent with iSTS gold chunks.

(a) PP + NP => PP

[2]Our chunker is available at http://semanticsimilarity.org

[3]http://www.cnts.ua.ac.be/conll2000/chunking/

[4]https://taku910.github.io/crfpp/

Table 6.3: Accuracies of OpenNLP chunker and our CRF chunker at chunk level (CL) and at sentence level (SL).

DataSet	Chunker	CL	SL
Headlines	O-NLP	53.88	16.13
	CRF	**83.32**	**63.23**
Image	O-NLP	52.71	5.33
	CRF	**90.29**	**74.93**

(b) VP + PRT => VP

(c) NP + CC + NP => NP

For example, it merges chunks [on] and [Friday] to form single PP chunk [on Friday] using rule (a).

The details about the chunking tool is available in (Maharjan et al., 2016).

We also chunked the input texts using the Open-NLP chunking tool (O-NLP). The results in SemEval 2015 test set which consisted of 375 Images data and 378 pairs of Headlines text are presented in Table 6.3. The accuracies were calculated at chunk level (*CL*) and sentence level (*SL*) by comparing the chunks created by the system against the manually created chunks (i.e., gold chunks). Since the accuracy of our CRF based chunker is better than that of OpenNLP, we used our CRF chunker for chunking texts in sys chunks category.

6.5 Chunk Alignment, Relation and Similarity Prediction

For a given sentence pair, the chunks of the first sentence were mapped to those from the second by assigning different semantic relations and scores based on a set of rules, similarity functions, and lookup resources. Before preforming alignments, we preprocessed texts as described in Section 6.3.

We built upon our previous system called NeRoSim. We refer the reader to Banjade, Niraula, et al. (2015) for further details of NeRoSim and in this section we describe our recent updates and results only. The limitation of NeRoSim was that the alignments were restricted to 1:1. We modified it to support many to many

alignments as well. Also, the NeRoSim system was able to process only gold chunks (i.e., chunks provided by the organizers). Now, the system can take input in the form of plain texts as well and create chunks on the fly. In addition to the chunking feature described in Section 6.4, the updates made to the system are described below.

Many-to-Many Alignments:

MULTI1: If there is any ALIC chunk (i.e., chunk which does not have any corresponding chunk in the other sentence because of the 1:1 alignment restriction) in sentence A whose content words are subsumed by the content words of any already aligned chunk (C) in another sentence B, merge ALIC chunk with the chunk in A paired with C. If the content words of merged chunk and those of C are same/equal, realign chunk C with merged chunk as EQUI and update the score to 5.0.

For example:

// [Iran] [hopes] [nuclear talks] [. . .].

// [Iran Nuclear Talks] [spur] [. . .].

Step 1:

nuclear talks <=> Iran Nuclear Talks // SPE2

Iran <=> //ALIC

Step2:

Iran nuclear talks <=> Iran Nuclear Talks // EQUI

MULTI2: In *MULTI1*, if all the content words of merged chunk and those of C are not matching completely, then realign chunk C with merged chunk but keep the previous alignment type and score.

Furthermore, we have expanded the rules for SIMI and EQUI.

EQx: If unmatched words are morphological inflections of each other and all other words in the chunks are already matched, assign the EQUI relation.

E.g. *Korean Air <=> Air Korea*

SIMIx: If nouns are matching but not the adjective or vice-versa, assign SIMI label.

E.g. *red carpet <=> brown carpet*

6.6 Experiments and Results

6.6.1 Runs

We run our system in three different settings (called Runs) which are described below.

***Run1*:** We included many-to-many alignment in NeRoSim (i.e., MULTI1 and MULTI2 were added).

***Run2*:** Same as Run1 in alignment but the alignment scores were assigned based on the average scores for each alignment type in the full training data.

***Run3*:** Same as Run2 but SIMIx and EQx rules added.

It should be noted that our system can process both chunked text and plain text. The results of our system using manually created chunks (*gold chunks*) as well as system created chunks (*sys chunks*) are presented in the next section.

6.6.2 Evaluation Method

The results were evaluated by calculating F1 scores based on Melamed (1998) which has been adopted in Agirre et al. (2016). This evaluation approach was proposed in the context of alignment for Machine Translation (MT) evaluation. In our case, the system generated alignments are compared with human annotated alignments.

Given,

g = gold standard token:token alignments (produced aligning all tokens in chunk:chunk alignments).

s = system token:token alignments (produced aligning all tokens in

chunk:chunk alignments).

$$precision(g, s) = \frac{overlap(g, s)}{|s|}$$
$$recall(g, s) = \frac{overlap(g, s)}{|g|}$$

where *overlap* returns the number of token:token alignments in common between both sets. The punctuations were ignored during the evaluation. In order to adjust the length of the chunks, the alignment at word level is counted and the weight is assigned as,

$$weight(t_1, t_2) = \frac{1}{max(fanout(t_1), fanout(t_2)}$$

where t_1 and t_2 are aligned tokens, and the $fanout(t)$ is the number of token:token alignments which have their origin at t.

In order to calculate the F1 score for the similarity score match, the weight of the token alignment is penalized for differences in score between the system generated score and the gold standard score as,

$$penalty = 1 - \frac{abs(score(t_1 : t_2, sys) - score(t_1 : t_2, gold))}{5}$$
$$weight(t_1, t_2) = \frac{1}{max(fanout(t_1), fanout(t_2)} * penalty$$

The precision, recall, and F1 scores are computed for all alignments of all pairs in one go (i.e., as opposed to averaging F1 of each sentence pair). Additional information about the evaluation method can be found in Agirre et al. (2016).

6.6.3 Results

The results on test datasets are presented in Table 6.4. The results are presented in terms of F1 scores on test set with gold chunks and sys generated chunks (separated

by /). The baseline system consists of several procedures as described in Agirre et al. (2016) and is considered as quite strong system in itself.

In *gold chunks* category, we can see that the alignment scores are higher compared to the baseline system and are very close to the best results from all submissions in those categories. However, the alignment type score in each case is relatively lower than the alignment-only score and it ultimately impacted the F1 score calculated for type and score together (i.e., T+S). We found the same pattern in all submitted systems in SemEval (Agirre et al., 2016). It indicates that the system's overall performance will be improved greatly if improvements can be made in predicting the alignment types. Also, the scores for student-answers are lower than headlines and image texts and it requires further analysis to fully understand why this is the case. One of the reasons might be that we did not use this dataset while developing the system and no prior information about such dataset was modeled. Additionally, more errors might have been introduced in our NLP pipeline as the texts in this dataset were not standard written texts compared to news headlines and image captions.

Similar to the results in gold chunks category, the values in Table 6.4 after / are the results on the test set but this time with *system chunks*. In image and headlines data, our system obtained the best results. However, following the same pattern as in gold chunk results, the F1 scores for alignments are high but the scores for predicting the alignment types are relatively lower. Also, the overall results in sys chunks category are lower than those of gold chunks. The reduction in the performance can be partly attributed to the error in chunking which is propagated to the final results.

In addition to the difficulty of the task of aligning the chunks and assigning relation types, we found some discrepancies in the annotation which we think induced some errors. For example, *on a sofa* <=> *on a blue sofa* (#65 in image

Table 6.4: F1 scores for chunk alignment, relation and similarity score prediction on test data with *gold chunks* and with *sys chunks* (separated by /). *Best* score is the highest score for each metric given by any of the participating systems in the shared task including the system submitted by the team involved in organizing the task.

System	Alignment	Relation Type	Sim. Score	Type+Score
		Headlines		
Baseline	0.8462/0.6486	0.5462/0.4379	0.7610/0.5912	0.5461/0.4379
Run1	0.9072/0.8366	0.6650/0.5605	0.8187/0.7394	0.6385/0.5384
Run2	0.9072/0.8366	0.6650/0.5605	0.836/0.7595	**0.6487/0.5467**
Run3	0.9072/0.8376	0.6583/0.5595	0.8329/0.7586	0.6405/0.5446
Best	0.9278/0.8366	0.7031/0.5605	0.8382/0.7595	0.6960/0.5467
		Image		
Baseline	0.8556/0.7127	0.4799/0.4043	0.7456/0.6251	0.4799/0.4043
Run1	0.8766/0.8429	0.6530/0.6148	0.7955/0.7591	0.6238/0.5870
Run2	0.8766/0.8429	0.6530/0.6148	0.8144/0.7806	0.6362/0.5990
Run3	0.8766/0.8429	0.6675/0.6276	0.8156/0.7813	**0.6483/0.6095**
Best	0.9077/0.8557	0.6867/0.6276	0.8552/0.7961	0.6708/0.6095
		Student-answers		
Baseline	0.8203/0.6188	0.5566/0.4431	0.7464/0.5702	0.5566/0.4431
Run1	0.8584/0.8165	0.5552/0.5157	0.7686/0.7248	0.5432/0.5049
Run2	0.8584/0.8165	0.5552/0.5157	0.7809/0.7367	**0.5458/0.5074**
Run3	0.8614/0.8181	0.5468/0.5112	0.7798/0.7360	0.5374/0.5029
Best	0.8922/0.8166	0.6511/0.5651	0.8433/0.7589	0.6385/0.5547

data), the human annotated label is SIMI but arguably the SPE2 label best describes the relation. Similarly, *in a field <=> in a green field* (#693 in image data), the SPE1 label has been found in the training set but it should be SPE2. In another example (#193 in image data), *A young boy <=> A young blonde girl* has been assigned a label SPE2 in the training data. Though the second chunk gives some additional details, the question is whether we should really compare them (and decide which one is more specific) because these two chunks are referring to different objects and therefore it sounds more like comparing apples and oranges.

6.7 Conclusion

In this chapter, we have presented our approach towards making the semantic similarity assessment more interpretable. In specific, we aligned chunks in the given pair of sentences and assigned semantic labels and similarity scores. Our system can

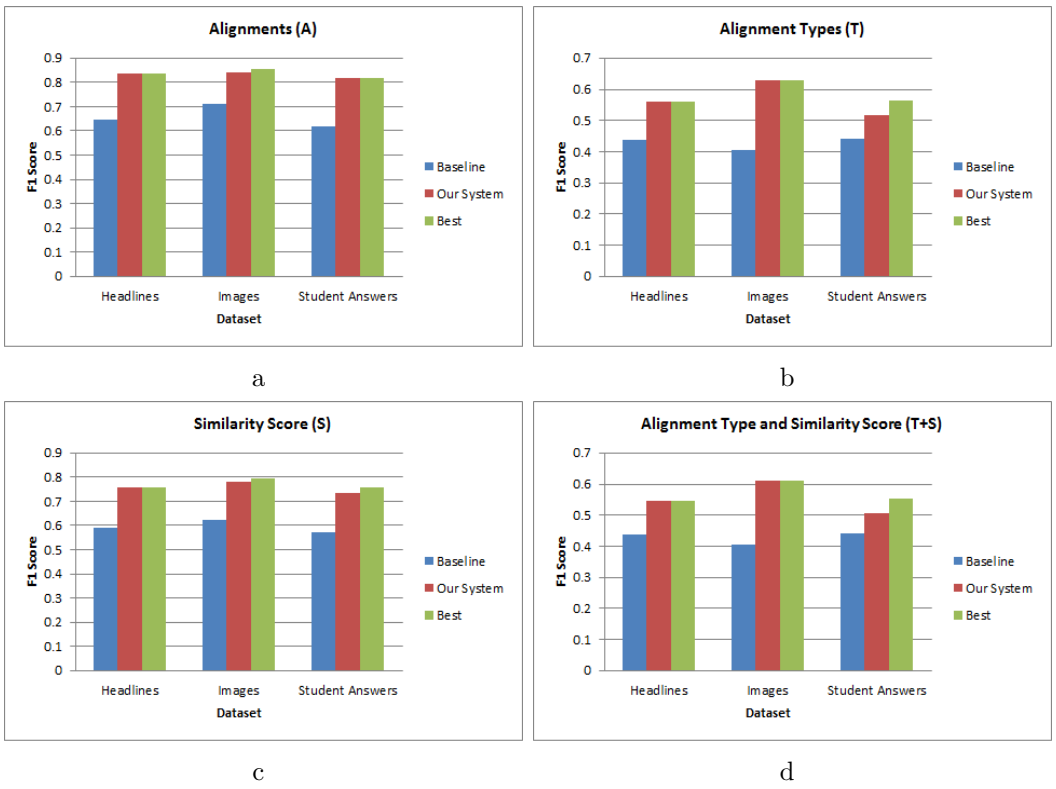


Fig. 6.1: Results on *sys chunks* category compared to baseline model and the best results among the participating submissions in SemEval 2016.

process chunked text as well as plain text input. In the later case, it itself creates chunks using our CRF based chunking tool. Our results on gold chunks as well as on system chunks categories are superior to the baseline results. In fact, our system was one of the best systems submitted in the SemEval shared task in 2016 and in overall, it performed the best.

The interpretable approach not only allows us to better understand the semantic similarity between texts but it can also be used in applications. For instance, the application we target to build using this approach is a feedback generator in our automatic answer assessment system and for following up question generation in conversational tutoring system. However, as our results suggested, the alignment in student answers is more difficult than in the more formal text. Particularly, assigning alignment labels requires further attention which we intend

to address in the future work. Furthermore, the annotated dataset is now quite big and it will certainly be useful in applying alternative approaches to predict chunk alignments and alignment types.

Chapter 7

Conclusion and Future Work

In this dissertation we have proposed several methods and created resources to improve measuring semantic similarity between short texts (word- and sentence-level) and automatic answer assessment. The target application is the dialogue based intelligent tutoring systems.

For measuring semantic similarity between given pair of sentences, we proposed two different approaches. In one of the approaches, we built Support Vector Regression models with various features and evaluated in SemEval 2016 evaluation dataset. We achieved correlation between our system's output and the human judgments up to 0.83 (0.735 in average) and our system emerged as one of the top performing systems submitted in the SemEval for the last three years. Our most recent system (which has not been reported in this dissertation) was ranked second among dozens of participating systems in SemEval competition in 2017. Similarly, our interpretable similarity methods achieved better results than strong baseline model and most of the participating systems in SemEval competition securing top position in overall evaluation. These results on SemEval datasets which are generally used for benchmarking semantic similarity models show our significant contribution in this research area. We also proposed Bayesian approach to model the semantic similarity and our domain adaptation models using transfer leaning approach has been found to be useful, particularly when the domain specific data is small in size.

We also addressed several issues in semantic similarity systems. One of the problems we often encounter is the missing word representations in vector based word representation models. We have proposed a Neural Network based mapping approach which can map vectors from one model where representation of the given

word is available (source) to another model where the representation for the word is missing (target). Our results with three popular pre-trained models show a significant growth in model coverage. The coverage increased from few to several times depending on the which model's vocabulary is taken as a reference. Also, our experiments show that the transformed vectors are well correlated to native target model's vectors and the mapped vectors can be used with confidence to augment the model of our interest.

Moreover, we proposed methods and dataset for negation handling in tutorial dialogues. We developed intra- and inter-sentential negation scope and focus detection models using Conditional Random Fields and the accuracy of our models detecting negation scope and focus was above 90%. We also created a dataset of negation in tutorial dialogues annotated for negation scope and focus in dialogue context which is very different from negation in literary style text.

As the target application of our work is the intelligent tutoring systems, we proposed models and created dataset to improve the open-ended answer assessment models. We used semantic similarity methods to answer assessment and augmented with additional knowledge, such as question difficulty and knowledge level of the students. Our Probabilistic Soft Logic (PSL) model including the semantic similarity and additional knowledge achieved the answer assessment accuracy of 57% when evaluated using DT-Grade dataset which includes the quite difficult cases to evaluate using semantic similarity methods alone. The answer assessment accuracy including additional knowledge to semantic similarity model is improved by about 7% compared to the model created using semantic similarity features only. This is particularly important because the students' responses in conversational tutoring systems vary a lot and are sometimes difficult to evaluate only using the semantic similarity information. We also created DT-Grade dataset which contains 900 responses collected during an experiment with DeepTutor tutoring system and

annotated for their correctness (correct, correct but incomplete, and incorrect) in the context of tutorial dialogue.

Additionally, we have proposed other models and tools for measuring the semantic similarity at word-level as well as at sentence-level that are not described in much details in this dissertation. For example, we contributed significantly in developing SEMILAR (a Semantic Similarity Toolkit) which has been used widely, created Latent Semantic Analysis (LSA) models from whole English Wikipedia articles and made available for download, and so on.

Future Work

In order to further advance the research in semantic similarity and answer assessment, we intend to work on one or more of the following areas.

- We have found that our model is giving higher scores where the expected similarity scores are around zero. Similarly, we intend to further analyze the performance of our models on various subsets, such as texts with high lexical overlap, low lexical overlap, high similarity, and low similarity and try to nail down the particular situations where our model is weak and address those issues.

- Most of the research on representation learning has been done on learning word or phrase representations and we also focused more on that level. Research on obtaining sentence representations has gained great attention in recent years and we also found sentence representations very effective for semantic similarity assessment. Therefore, in addition to learning word representations and linearly combining them to get the sentence representations we intend to explore on learning sentence representations using more direct approach.

- Perform word vector mapping experiments with additional word representation models and try to understand which types of models are most effective in mapping. Also, extend the vector based word representation mapping approach to phrases and sentence level which are even more sparser than words.

- Integrate the negation scope and focus detection models in Natural Language Understanding (NLU) applications including semantic similarity and answer assessment.

- Improve rules weight learning in Probabilistic Soft Logic (PSL) model using more heuristics in situations where weight learning from the full dataset is not computationally feasible. Clustering data and refactoring the graphical model to further reduce the scale of the problem can be one direction to take.

- The interpretable approach not only allows us to better understand the semantic similarity between texts, it also can be used in applications. For instance, the application we target to build using this approach is a feedback generator in our automatic answer assessment system and for follow up question generation in conversational tutoring systems. However, as our results suggested, the alignment in student answers is more difficult than in the more formal text. Particularly, assigning alignment labels requires further attention which we intend to address in the future work. Furthermore, we intend to annotate student answers collected in tutoring system and align with reference answers and develop model for detail feedback generation.

References

Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., ... Wiebe, J. (2014). Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (semeval 2014)* (pp. 81–91).

Agirre, E., Banea, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Mihalcea, R., ... Wiebe, J. (2016). Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. *Proceedings of SemEval*, 497–511.

Agirre, E., Baneab, C., Cardiec, C., Cerd, D., Diabe, M., Gonzalez-Agirrea, A., ... others (2015). Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (semeval 2015)* (pp. 252–263).

Agirre, E., Diab, M., Cer, D., & Gonzalez-Agirre, A. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the first joint conference on lexical and computational semantics-volume 1: Proceedings of the main conference and the shared task, and volume 2: Proceedings of the sixth international workshop on semantic evaluation* (pp. 385–393).

Alexandrescu, A., & Kirchhoff, K. (2006). Factored neural language models. In *Proceedings of the human language technology conference of the naacl, companion volume: Short papers* (pp. 1–4).

Anand, P., & Martell, C. (2012). Annotating the focus of negation in terms of questions under discussion. In *Proceedings of the workshop on extra-propositional aspects of meaning in computational linguistics* (pp. 65–69).

Androutsopoulos, I., & Malakasiotis, P. (2010). A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, *38*, 135–187.

Bach, S., Huang, B., London, B., & Getoor, L. (2013). Hinge-loss markov random fields: Convex inference for structured prediction. *arXiv preprint arXiv:1309.6813*.

Bach, S. H., Broecheler, M., Huang, B., & Getoor, L. (2015). Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406*.

Banjade, R., Maharjan, N., Gautam, D., & Rus, V. (2016). Dtsim at semeval-2016 task 1: Semantic similarity model including multi-level alignment and vector-based compositional semantics. *Proceedings of SemEval*, 640–644.

Banjade, R., Maharjan, N., Gautam, D., & Rus, V. (2017). Pooling word representations across models. *Proceedings of 18th International Conference on Computational Linguistics and Intelligent Text Processing*.

Banjade, R., Maharjan, N., Niraula, N. B., Gautam, D., Samei, B., & Rus, V. (2016). Evaluation dataset (dt-grade) and word weighting approach towards constructed short answers assessment in tutorial dialogue context.

Banjade, R., Maharjan, N., Niraula, N. B., & Rus, V. (2016). Dtsim at semeval-2016 task 2: Interpreting similarity of texts based on automated chunking, chunk alignment and semantic relation prediction. *Proceedings of SemEval*, 809–813.

Banjade, R., Maharjan, N., Niraula, N. B., Rus, V., & Gautam, D. (2015). Lemon and tea are not similar: Measuring word-to-word similarity by combining different methods. In *International conference on intelligent text processing and computational linguistics* (pp. 335–346).

Banjade, R., Niraula, N. B., Maharjan, N., Rus, V., Stefanescu, D., Lintean, M., & Gautam, D. (2015). Nerosim: A system for measuring and interpreting semantic textual similarity. *SemEval-2015*, 164.

Banjade, R., Niraula, N. B., & Rus, V. (2016). Towards detecting intra-and inter-sentential negation scope and focus in dialogue. In *The twenty-ninth*

international flairs conference.

Banjade, R., & Rus, V. (2016). Dt-neg: Tutorial dialogues annotated for negation scope and focus in context..

Baroni, M., Dinu, G., & Kruszewski, G. (2014). Dont count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd annual meeting of the association for computational linguistics* (Vol. 1, pp. 238–247).

Baroni, M., & Zamparelli, R. (2010). Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 1183–1193).

Basu, S., Jacobs, C., & Vanderwende, L. (2013). Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, *1*, 391–402.

Batchkarov, M., Kober, T., Reffin, J., Weeds, J., & Weir, D. (2016). A critique of word similarity as a method for evaluating distributional semantic models. *ACL 2016*, 7.

Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, *3*, 1137–1155.

Blanco, E., & Moldovan, D. (2012). Fine-grained focus for pinpointing positive implicit meaning from negated statements. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 456–465).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational researcher*, *13*(6),

4–16.

Brockett, C. (2007). Aligning the rte 2006 corpus. *Microsoft Research*.

Brockett, C., & Dolan, W. B. (2005). Support vector machines for paraphrase identification and corpus construction. In *Proceedings of the 3rd international workshop on paraphrasing* (pp. 1–8).

Burgess, C., & Lund, K. (1995). Hyperspace analog to language (hal): A general model of semantic representation. In *Proceedings of the annual meeting of the psychonomic society* (Vol. 12, pp. 177–210).

Carberry, S. (1989). A pragmatics-based approach to ellipsis resolution. *Computational Linguistics*, *15*(2), 75–96.

Carbonell, J. G. (1983). Discourse pragmatics and ellipsis resolution in task-oriented natural language interfaces. In *Proceedings of the 21st annual meeting on association for computational linguistics* (pp. 164–168).

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

Chapman, W. W., Bridewell, W., Hanbury, P., Cooper, G. F., & Buchanan, B. G. (2001). Evaluation of negation phrases in narrative clinical reports. In *Proceedings of the amia symposium* (p. 105).

Choi, Y., & Cardie, C. (2008). Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 793–801).

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, *12*, 2493–2537.

Corley, C., & Mihalcea, R. (2005). Measuring the semantic similarity of texts. In *Proceedings of the acl workshop on empirical modeling of semantic equivalence and entailment* (pp. 13–18).

Councill, I. G., McDonald, R., & Velikovich, L. (2010). What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing* (pp. 51–59).

DMello, S., Lehman, B., Pekrun, R., & Graesser, A. (2014). Confusion can be beneficial for learning. *Learning and Instruction*, *29*, 153–170.

Dolan, B., Quirk, C., & Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on computational linguistics* (p. 350).

Dzikovska, M. O., Moore, J. D., Steinhauser, N., Campbell, G., Farrow, E., & Callaway, C. B. (2010). Beetle ii: a system for tutoring and computational linguistics experimentation. In *Proceedings of the acl 2010 system demonstrations* (pp. 13–18).

Dzikovska, M. O., Nielsen, R. D., Brew, C., Leacock, C., Giampiccolo, D., Bentivogli, L., ... Dang, H. T. (2013). *Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge* (Tech. Rep.). DTIC Document.

Elkin, P. L., Brown, S. H., Bauer, B. A., Husser, C. S., Carruth, W., Bergstrom, L. R., & Wahner-Roedler, D. L. (2005). A controlled trial of automated classification of negation from clinical notes. *BMC medical informatics and decision making*, *5*(1), 13.

Evens, M., & Michael, J. (2006). One-on-one tutoring by humans and machines. *Computer Science Department, Illinois Institute of Technology*.

Fernando, S., & Stevenson, M. (2008). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th annual research colloquium of the uk special interest group for computational linguistics* (pp. 45–52).

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., &

Ruppin, E. (2001). Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on world wide web* (pp. 406–414).

Gabrilovich, E., & Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Ijcai* (Vol. 7, pp. 1606–1611).

Gao, J., Deng, L., Gamon, M., He, X., & Pantel, P. (2014, June 13). *Modeling interestingness with deep neural networks.* Google Patents. (US Patent App. 14/304,863)

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical science*, 457–472.

Gindl, S., Kaiser, K., & Miksch, S. (2008). Syntactical negation detection in clinical practice guidelines. *Studies in health technology and informatics*, *136*, 187.

Graesser, A. C., Chipman, P., Haynes, B. C., & Olney, A. (2005). Autotutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, *48*(4), 612–618.

Graesser, A. C., VanLehn, K., Rosé, C. P., Jordan, P. W., & Harter, D. (2001). Intelligent tutoring systems with conversational dialogue. *AI magazine*, *22*(4), 39.

Graesser, A. C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Tutoring Research Group, T. R. G., & Person, N. (2000). Using latent semantic analysis to evaluate the contributions of students in autotutor. *Interactive learning environments*, *8*(2), 129–147.

Hall, M. A., & Smith, L. A. (1998). Practical feature subset selection for machine learning.

Han, L., Kashyap, A., Finin, T., Mayfield, J., & Weese, J. (2013). Umbc ebiquity-core: Semantic textual similarity systems. In *Proceedings of the second joint conference on lexical and computational semantics* (Vol. 1, pp.

44–52).

High, R. (2012). The era of cognitive systems: An inside look at ibm watson and how it works. *IBM Corporation, Redbooks*.

Hill, F., Reichart, R., & Korhonen, A. (2014). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *arXiv preprint arXiv:1408.3456*.

Horn, L. (1989). A natural history of negation.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd acm international conference on conference on information & knowledge management* (pp. 2333–2338).

Huddleston, R., Pullum, G. K., et al. (2002). The cambridge grammar of english. *Language. Cambridge: Cambridge University Press*, 1–23.

Iacobacci, I., Pilehvar, M. T., & Navigli, R. (2015). Sensembed: learning sense embeddings for word and relational similarity. In *Proceedings of acl* (pp. 95–105).

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

Kimmig, A., Bach, S., Broecheler, M., Huang, B., & Getoor, L. (2012). A short introduction to probabilistic soft logic. In *Proceedings of the nips workshop on probabilistic programming: Foundations and applications* (pp. 1–4).

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. In *Advances in neural information processing systems* (pp. 3294–3302).

Konstantinova, N., De Sousa, S. C., & Sheila, J. (2011). Annotating negation and speculation: the case of the review domain. In *Ranlp student research workshop* (pp. 139–144).

Kruschke, J. (2014). *Doing bayesian data analysis: A tutorial with r, jags, and stan*.

Academic Press.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, *2*(1-2), 83–97.

Kulkarni, C., Wei, K. P., Le, H., Chia, D., Papadopoulos, K., Cheng, J., ... Klemmer, S. R. (2015). Peer and self assessment in massive online classes. In *Design thinking research* (pp. 131–168). Springer.

Landauer, T. K. (2003). Automatic essay assessment. *Assessment in education: Principles, policy & practice*, *10*(3), 295–308.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, *25*(2-3), 259–284.

Leacock, C., & Chodorow, M. (2003). C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, *37*(4), 389–405.

Lei, T., Xin, Y., Zhang, Y., Barzilay, R., & Jaakkola, T. (2014). Low-rank tensors for scoring dependency structures. In *Proceedings of the 52nd annual meeting of the association for computational linguistics* (Vol. 1, pp. 1381–1391).

Li, Y., Bandar, Z. A., & McLean, D. (2003). An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on knowledge and data engineering*, *15*(4), 871–882.

Li, Y. H., & Jain, A. K. (1998). Classification of text documents. *The Computer Journal*, *41*(8), 537–546.

Lin, D., et al. (1998). An information-theoretic definition of similarity. In *Icml* (Vol. 98, pp. 296–304).

Lintean, M., & Rus, V. (2015). An optimal quadratic approach to monolingual paraphrase alignment. In *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, 2015, vilnius, lithuania* (pp. 127–134).

Lintean, M. C., Moldovan, C., Rus, V., & McNamara, D. S. (2010). The role of

local and global weighting in assessing the semantic similarity of texts using latent semantic analysis. In *Flairs conference* (pp. 235–240).

Lintean, M. C., & Rus, V. (2011). Dissimilarity kernels for paraphrase identification. In *Flairs conference.*

Lunn, D., Spiegelhalter, D., Thomas, A., & Best, N. (2009). The bugs project: Evolution, critique and future directions. *Statistics in Medicine*, *28*(25), 3049–3067.

Luong, M.-T., Socher, R., & Manning, C. D. (2013). Better word representations with recursive neural networks for morphology. *CoNLL-2013*, *104*.

Mac Namee, B., Kelleher, J., & Delany, S. J. (2008). Medical language processing for patient diagnosis using text classification and negation labelling.

Madnani, N., Tetreault, J., & Chodorow, M. (2012). Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 182–190).

Maharjan, N., Banjade, R., Gautam, D., J. Tamang, L., & Rus, V. (2017). Dt_team at semeval-2017 task 1: Semantic similarity using alignments, sentence-level embeddings and gaussian mixture model output. In *Semeval* (p. 120124).

Maharjan, N., Banjade, R., Niraula, N., & Rus, V. (2016). Semaligner: A tool for aligning chunks with semantic relation types and semantic similarity scores. In *Lrec.*

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval* (Vol. 1). Cambridge university press Cambridge.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *Acl (system demonstrations)* (pp. 55–60).

Martin, J., & VanLehn, K. (1995). Student assessment using bayesian nets.

International Journal of Human-Computer Studies, *42*(6), 575–591.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit. Retrieved 2013-06-01, from `http://mallet.cs.umass.edu`.

McNamara, D. S., Levinstein, I. B., & Boonthum, C. (2004). istart: Interactive strategy training for active reading and thinking. *Behavior Research Methods, Instruments, & Computers*, *36*(2), 222–233.

Melamed, I. D. (1998). Manual annotation of translational equivalence: The blinker project. *arXiv preprint cmp-lg/9805005*.

Miestamo, M. (2006). On the complexity of standard negation. *A man of measure: Festschrift in Honour of Fred Karlsson on His 60th Birthday*, 345–356.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Mohler, M., Bunescu, R., & Mihalcea, R. (2011). Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1* (pp. 752–762).

Mohler, M., & Mihalcea, R. (2009). Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th conference of the european chapter of the association for computational linguistics* (pp. 567–575).

Moilanen, K., & Pulman, S. (2007). Sentiment composition. In *Proceedings of ranlp* (Vol. 7, pp. 378–382).

Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, *6*(4), 525–533.

Morante, R., & Blanco, E. (2012). * sem 2012 shared task: Resolving the scope and focus of negation. In *Proceedings of the first joint conference on lexical and computational semantics-volume 1: Proceedings of the main conference and the shared task, and volume 2: Proceedings of the sixth international workshop on semantic evaluation* (pp. 265–274).

Morante, R., & Daelemans, W. (2009). Learning the scope of hedge cues in biomedical texts. In *Proceedings of the workshop on current trends in biomedical natural language processing* (pp. 28–36).

Morante, R., Schrauwen, S., & Daelemans, W. (2011). Annotation of negation cues and their scope: Guidelines v1. *Computational linguistics and psycholinguistics technical report series, CTRS-003*.

Murrugarra, N., Lu, S., & Li, M. (2013). Automatic grading student answers.

Mutalik, P. G., Deshpande, A., & Nadkarni, P. M. (2001). Use of general-purpose negation detection to augment concept indexing of medical documents. *Journal of the American Medical Informatics Association*, *8*(6), 598–609.

Nayak, N., Angeli, G., & Manning, C. D. (2016). Evaluating word embeddings using a representative suite of practical tasks. *ACL 2016*, 19.

Nenkova, A., & McKeown, K. (2012). A survey of text summarization techniques. In *Mining text data* (pp. 43–76). Springer.

Nielsen, R. D., Ward, W., Martin, J. H., & Palmer, M. (2008). Annotating students' understanding of science concepts. In *Lrec.*

Niraula, N. B., Gautam, D., Banjade, R., Maharjan, N., & Rus, V. (2015). Combining word representations for measuring word relatedness and similarity. In *Flairs conference* (pp. 199–204).

Niraula, N. B., Rus, V., Banjade, R., Stefanescu, D., Baggett, W., & Morgan, B.

(2014). The dare corpus: A resource for anaphora resolution in dialogue based intelligent tutoring systems. In *Lrec* (pp. 3199–3203).

Olney, A., DMello, S., Person, N., Cade, W., Hays, P., Williams, C., ... Graesser, A. (2012). Guru: A computer tutor that models expert human tutors. In *Intelligent tutoring systems* (pp. 256–261).

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 311–318).

Pavlick, E., Bos, J., Nissim, M., Beller, C., Van Durme, B., & Callison-Burch, C. (2015). Ppdb 2.0: Better paraphrase ranking, finegrained entailment relations, word embeddings, and style classification. In *Proc. acl.*

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at hlt-naacl 2004* (pp. 38–41).

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, *12*, 1532–1543.

Pérez, D., Gliozzo, A. M., Strapparava, C., Alfonseca, E., Rodríguez, P., & Magnini, B. (2005). Automatic assessment of students' free-text answers underpinned by the combination of a bleu-inspired algorithm and latent semantic analysis. In *Flairs conference* (pp. 358–363).

Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., & Manning, C. (2010). A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 492–501).

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

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine learning*, *62*(1-2), 107–136.

Rokach, L., Romano, R., & Maimon, O. (2008). Negation recognition in medical narrative reports. *Information Retrieval*, *11*(6), 499–538.

Rooth, M. (1996). *Focus. the handbook of contemporary semantic theory, ed. by shalom lappin, 271-97.* Oxford: Blackwell.

Rosenberg, S., & Bergler, S. (2012). Uconcordia: Clac negation focus detection at* sem 2012. In *Proceedings of the first joint conference on lexical and computational semantics-volume 1: Proceedings of the main conference and the shared task, and volume 2: Proceedings of the sixth international workshop on semantic evaluation* (pp. 294–300).

Rus, V., Banjade, R., & Lintean, M. C. (2014). On paraphrase identification corpora. In *Lrec* (pp. 2422–2429).

Rus, V., DMello, S., Hu, X., & Graesser, A. (2013). Recent advances in conversational intelligent tutoring systems. *AI magazine*, *34*(3), 42–54.

Rus, V., & Graesser, A. C. (2006). Deeper natural language processing for evaluating student answers in intelligent tutoring systems. In *Proceedings of the national conference on artificial intelligence* (Vol. 21, p. 1495).

Rus, V., & Lintean, M. (2012). A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the seventh workshop on building educational applications using nlp* (pp. 157–162).

Rus, V., Lintean, M., Moldovan, C., Baggett, W., Niraula, N., & Morgan, B. (2012). The similar corpus: A resource to foster the qualitative understanding of semantic similarity of texts. In *Semantic relations ii: Enhancing resources*

and applications, the 8th language resources and evaluation conference (lrec 2012), may (pp. 23–25).

Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013). Semilar: The semantic similarity toolkit. In *Acl (conference system demonstrations)* (pp. 163–168).

Rus, V., Niraula, N., & Banjade, R. (2013). Similarity measures based on latent dirichlet allocation. In *International conference on intelligent text processing and computational linguistics* (pp. 459–470).

Rus, V., Niraula, N., & Banjade, R. (2015). Deeptutor: An effective, online intelligent tutoring system that promotes deep learning. In *Twenty-ninth aaai conference on artificial intelligence.*

Rychalska, B., Pakulska, K., Chodorowska, K., Walczak, W., & Andruszkiewicz, P. (2016). Samsung poland nlp team at semeval-2016 task 1: Necessity for diversity; combining recursive autoencoders, wordnet and ensemble methods to measure semantic similarity. In *Proceedings of the 10th international workshop on semantic evaluation (semeval 2016), san diego, ca, usa.*

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513–523.

Severyn, A., Nicosia, M., & Moschitti, A. (2013). ikernels-core: Tree kernel learning for textual similarity. In *Proceedings of the second joint conference on lexical and computational semantics* (Vol. 1, pp. 53–58).

Shen, Y., He, X., Gao, J., Deng, L., & Mesnil, G. (2014). A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd acm international conference on conference on information and knowledge management* (pp. 101–110).

Shrestha, P., & Solorio, T. (2015). Identification of original document by using textual similarities. In *International conference on intelligent text processing*

and computational linguistics (pp. 643–654).

Shute, V. J. (2008). Focus on formative feedback. *Review of educational research*, *78*(1), 153–189.

Socher, R., Huang, E. H., Pennin, J., Manning, C. D., & Ng, A. Y. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems* (pp. 801–809).

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (emnlp)* (Vol. 1631, p. 1642).

Stefanescu, D., Banjade, R., & Rus, V. (2014a). A sentence similarity method based on chunking and information content. In *International conference on intelligent text processing and computational linguistics* (pp. 442–453).

Stefanescu, D., Banjade, R., & Rus, V. (2014b). Latent semantic analysis models on wikipedia and tasa.

Stefanescu, D., Rus, V., Niraula, N. B., & Banjade, R. (2014). Combining knowledge and corpus-b to-word similarity measures for word.

Sukkarieh, J. Z., & Blackmore, J. (2009). c-rater: Automatic content scoring for short constructed responses. In *Flairs conference* (pp. 290–295).

Sukkarieh, J. Z., & Bolge, E. (2010). Building a textual entailment suite for the evaluation of automatic content scoring technologies. In *Lrec.*

Sultan, M. A., Bethard, S., & Sumner, T. (2015). Dls@cu: Sentence similarity from word alignment and semantic vector composition. In *Proceedings of the 9th international workshop on semantic evaluation* (pp. 148–153).

Thompson, P., Nawaz, R., McNaught, J., & Ananiadou, S. (2011). Enriching a biomedical event corpus with meta-knowledge annotation. *BMC bioinformatics*, *12*(1), 393.

Tjong Kim Sang, E. F., & Buchholz, S. (2000). Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on learning language in logic and the 4th conference on computational natural language learning-volume 7* (pp. 127–132).

Tottie, G. (1993). Negation in english speech and writing: A study in variation. *Language*, *69*(3), 590–593.

Turian, J., Ratinov, L., & Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 384–394).

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, *59*(236), 433–460.

Turney, P. D. (2001). Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *European conference on machine learning* (pp. 491–502).

VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A., & Rosé, C. P. (2007). When are tutorial dialogues more effective than reading? *Cognitive science*, *31*(1), 3–62.

Vincze, V., Szarvas, G., Farkas, R., Móra, G., & Csirik, J. (2008). The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*, *9*(11), S9.

Wedin, M. V. (1990). Negation and quantification in aristotle. *History and Philosophy of Logic*, *11*(2), 131–150.

Williams, C. K., & Rasmussen, C. E. (1996). Gaussian processes for regression. *Advances in neural information processing systems*, 514–520.

Williams, J. D., Niraula, N. B., Dasigi, P., Lakshmiratan, A., Suarez, C. G. J., Reddy, M., & Zweig, G. (2015). Rapidly scaling dialog systems with interactive learning. In *Natural language dialog systems and intelligent assistants* (pp. 1–13). Springer.

Wu, S. T.-I., Miller, T. A., Masanz, J. J., Coarr, M., Carrell, D., Halgrim, S. R., ... Clark, C. (2013). Negation's not solved: Reconsidering negation annotation and evaluation. In *Amia.*

Yu, M., & Dredze, M. (2014). Improving lexical embeddings with semantic knowledge. In *Association for computational linguistics (acl)* (pp. 545–550).

Zou, B., Zhou, G., & Zhu, Q. (2014). Negation focus identification with contextual discourse information. In *Acl (1)* (pp. 522–530).