

University of Memphis

University of Memphis Digital Commons

Electronic Theses and Dissertations

1-1-2019

Deeper Understanding of Tutorial Dialogues and Student Assessment

Nabin Nabin Maharjan

Follow this and additional works at: <https://digitalcommons.memphis.edu/etd>

Recommended Citation

Maharjan, Nabin Nabin, "Deeper Understanding of Tutorial Dialogues and Student Assessment" (2019). *Electronic Theses and Dissertations*. 2928.
<https://digitalcommons.memphis.edu/etd/2928>

This Dissertation is brought to you for free and open access by University of Memphis Digital Commons. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of University of Memphis Digital Commons. For more information, please contact khggerty@memphis.edu.

DEEPER UNDERSTANDING OF TUTORIAL DIALOGUES AND STUDENT
ASSESSMENT

by
Nabin Maharjan

A Dissertation
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

Major: Computer Science

The University of Memphis

April 15, 2019

©2019 Nabin Maharjan
All rights reserved

DEDICATION

To my parents and my wife.

ACKNOWLEDGMENTS

First, I am immensely grateful to my advisor and committee chair Dr. Vasile Rus for his great support and guidance towards completing my Ph.D. degree. I enjoyed my life as a Ph.D. student because of his genial and supportive personality. I am also grateful to the rest of my dissertation committee members Dr. Andrew McGregor Olney, Dr. Scott Fleming, and Dr. Deepak Venugopal for their helpful discussions and suggestions.

Also, I would like to acknowledge the financial support from the grant R305A100875 from the Institute for Education Sciences to Dr. Vasile Rus, the Institute for Intelligent Systems, the Institute for Intelligent Systems Student Organization, and the Graduate Student Association of the University of Memphis.

I am also thankful to my friends and colleagues including Dr. Rajendra Banjade, Dipesh Gautam and Dr. Nobal Niraula for their support and collaboration in research activities, which provided the foundation for my Ph.D. dissertation work.

I am also deeply indebted to my parents who motivated me to pursue the Ph.D. degree. I would also like to thank my brother and my in-laws. Lastly, I would like to thank my wife Rosita for her unwavering love and support.

ABSTRACT

Maharjan, Nabin Ph.D. The University of Memphis. April 15, 2019. Deeper Understanding of Tutorial Dialogues and Student Assessment. Major Professor: Vasile Rus, Ph.D.

Bloom (1984) reported two standard deviation improvement with human tutoring which inspired many researchers to develop Intelligent Tutoring Systems (ITSs) that are as effective as human tutoring. However, recent studies suggest that the 2-sigma result was misleading and that current ITSs are as good as human tutors. Nevertheless, we can think of 2 standard deviations as the benchmark for tutoring effectiveness of ideal expert tutors. In the case of ITSs, there is still the possibility that ITSs could be better than humans.

One way to improve the ITSs would be identifying, understanding, and then successfully implementing effective tutorial strategies that lead to learning gains. Another step towards improving the effectiveness of ITSs is an accurate assessment of student responses. However, evaluating student answers in tutorial dialogues is challenging. The student answers often refer to the entities in the previous dialogue turns and problem description. Therefore, the student answers should be evaluated by taking dialogue context into account. Moreover, the system should explain which parts of the student answer are correct and which are incorrect. Such explanation capability allows the ITSs to provide targeted feedback to help students reflect upon and correct their knowledge deficits. Furthermore, targeted feedback increases learners' engagement, enabling them to persist in solving the instructional task at hand on their own.

In this dissertation, we describe our approach to discover and understand effective tutorial strategies employed by effective human tutors while interacting with learners. We also present various approaches to automatically assess students' contributions using general methods that we developed for semantic analysis of short texts. We explain our work using generic semantic similarity approaches to

evaluate the semantic similarity between individual learner contributions and ideal answers provided by experts for target instructional tasks. We also describe our method to assess student performance based on tutorial dialogue context, accounting for linguistic phenomena such as ellipsis and pronouns. We then propose an approach to provide an explanatory capability for assessing student responses. Finally, we recommend a novel method based on concept maps for jointly evaluating and interpreting the correctness of student responses.

TABLE OF CONTENTS

Contents	Page
List of Figures	ix
List of Tables	xi
1 Introduction	1
1.1 Intelligent Tutoring Systems	1
1.1.1 Research Challenges	3
1.2 Goal	6
1.3 Research Questions	6
1.4 Summary of Primary Contributions	7
2 Discovering Effective Tutorial Strategies Employed by Professional Tutors	9
2.1 Related Work	11
2.2 Language as Action	11
2.3 Dialogue Taxonomy	12
2.4 Identifying Patterns in Effective Tutorial Sessions	15
2.5 Feature Selection	17
2.6 Experiments and Results	19
2.6.1 Data	19
2.6.2 Dialogue Classification	21
2.6.3 Tutorial Session Analysis	23
2.6.4 Analysis using human judgment measure of learning gain	24
2.6.5 Analysis using computer measure of learning gain	36
2.7 Conclusion	39
3 Assessing Semantic Textual Similarity	41
3.1 Related Work	43
3.1.1 Word-to-word Similarity	43
3.1.2 Sentence-to-Sentence Similarity	45
3.2 Preprocessing	48
3.3 Approach	48
3.3.1 Feature Generation	49
Word Similarity Methods	49
Sentence Similarity Methods	49
3.3.2 Feature Selection	54
3.4 Experiment and Results	56
3.4.1 Datasets	56
Training Data	56
Test Data	56
STS Benchmark Data	56
3.4.2 Models and Runs	57

3.4.3	Results and Discussions	58
3.5	Conclusion	62
4	Assessing Student Answers in Tutorial Dialogue Context	63
4.1	Related Work	66
4.2	Approach	68
4.2.1	Gaussian Mixture Model	68
4.2.2	The LSTM Approach	72
4.3	Experiments and Results	74
4.3.1	Data	74
4.3.2	GMM based Experiments and Results	75
4.3.3	LSTM based Experiments and Results	79
4.4	Conclusion	80
5	SemAligner: A Tool for Interpreting Semantic Textual Similarity	82
5.1	Related Work	84
5.2	The SemAligner tool	85
5.2.1	Chunker	85
5.2.2	Alignment System	88
5.2.3	Alignment Output	89
5.2.4	Alignment System Evaluation	90
5.3	Conclusion	92
6	A Concept Map based Assessment of Free Student Answers in Tutorial Dialogues	94
6.1	Related Work	100
6.2	Concept Map based Approach	103
6.2.1	Creation of Ideal Concept Maps	103
6.2.2	Automated Extraction of Student Concept Maps	103
6.2.3	Assessment System	107
6.3	Experiment and Results	110
6.3.1	Data	111
6.3.2	Ideal Concept Maps	111
6.3.3	Automated Tuple Quality Evaluation	113
6.3.4	Binary Classification Task	114
6.3.5	Multi-level Classification Task	116
6.4	Conclusion	118
7	Conclusion and Future Work	119
	References	125

LIST OF FIGURES

Figure	Page
1.1 Inner Loop Interface of DeepTutor tutoring system.	3
2.1 Tutorial Data Annotation.	19
2.2 Distribution of human ratings of Evidence of Learning (EL) and Evidence of Soundness (ES).	24
2.3 Distribution of dialogue act profile of top versus bottom 10% sessions. a show the profile for tutors only while b shows the profile for both tutors and students.	25
2.4 Distribution of mode switch profile of top versus bottom 10% sessions. a show the profile for tutors only while b shows the profile for both tutors and students.	26
2.5 Dialogue mode sequence logo for top 10% sessions up to average mode switch length of 21.	28
2.6 Dialogue mode sequence logo for bottom 10% sessions up to average mode switch length of 11.	29
2.7 Tutorial Markov Process for effective tutorial sessions.	33
2.8 Dialogue mode profiles of top versus bottom 25% sessions for tutors only.	37
2.9 Dialogue mode profiles of top versus bottom 25% sessions including both tutor and student initiated modes.	38
2.10 Dialogue mode sequence logo for top 25% sessions of average length 20.	39
2.11 Dialogue mode sequence logo for bottom 25% sessions of average length 19.	40
3.1 General Pipeline of DT_TEAM System.	46
3.2 R1 system output in evaluation data plotted against human judgments (in ascending order).	59
4.1 LSTM model architecture with tri-letter word encodings.	72
5.1 System pipeline of SemAligner tool.	86

6.1	A concept map based student answer assessment approach.	95
6.2	A snippet of an XML representation of an instructional task in Deep-Tutor.	96
6.3	An ideal concept map representation for the task shown in Figure 6.2.	97
6.4	A comparison of an ideal concept map (a) and computer-generated concept map (b) for the ideal answer: “ <i>When velocity is constant, the acceleration is zero; therefore the sum of the forces will equal zero</i> ”.	98
6.5	Annotating data for binary classification. TupleIds 2_6 and 2_7 consist of expectation id 2 concatenated with synsetId 6 and 7 respectively.	111

LIST OF TABLES

Table	Page
1.1 A sample question and answer between student and DeepTutor with the ideal expected answer.	5
2.1 A snippet of tutorial dialogue between student and tutor labeled with dialogue acts and modes. PK refers to Prior Knowledge.	12
2.2 Features	18
2.3 Average Inter Annotator Agreement Between Two Independent Annotators. Mode* and Mode** represent dialogue mode agreement between verifier and first annotator and, verifier and second annotator respectively.	21
2.4 Performance of dialogue act classifiers	22
2.5 Performance of dialogue act - subact classifiers. CRF-2 uses gold dialogue acts to analyze impact of noisy predicted acts.	22
2.6 Performance of dialogue mode classifier	23
2.7 Mapping of dialogue modes to symbols.	27
2.8 Top 12 discriminant speaker differentiated act subsequences.	30
2.9 Top 5 discriminant speaker differentiated act-subact subsequences.	33
2.10 Discriminant mode subsequences. Symbols in subsequences represent dialog modes as described in Table 2.7.	34
3.1 Interpretation of similarity score (SS) with corresponding example sentence pairs (Agirre et al., 2015).	42
3.2 Example of chunk alignment between two short text with semantic labels and scores.	50
3.3 Summary of training data.	56
3.4 Distribution of STS benchmark data according to different genres and data partitions.	57

3.5	Detailed breakdown of STS benchmark data by original names and task years of the datasets.	57
3.6	Results of our submitted runs on test data (1 st is the best result among the participants).	58
3.7	A set of highly correlated features with gold scores in test data.	59
3.8	Difficult English sentence pairs (Cer, Diab, Agirre, Lopez-Gazpio, & Specia, 2017).	60
3.9	Performances of top performing STS systems against difficult English sentence pairs (Cer et al., 2017).	61
3.10	Performances of participating STS systems on STS benchmark data (Cer et al., 2017).	61
4.1	An example of a problem and student answers to a tutor question. Context is needed to assess answers <i>A1-A3</i> properly.	64
4.2	Features used in model development.	70
4.3	Performances of different GMM models. Baseline model M0 is a logistic model presented by Banjade, Maharjan, Niraula, Gautam, et al. (2016). U-CNT refers to Unique Counts. <i>M14*</i> model was developed using instances not requiring contextual information alone and evaluated on instances requiring contextual information. <i>LSTM*</i> refers to our LSTM approach (Maharjan, Gautam, & Rus, 2018).	75
4.4	Correlation analysis between various features and GMM weights for correctness labels (CNT Contradictory, C - Correct, CBI Correct-but-incomplete, IC - Incorrect) computed using model M7. F1-F11 are the features from Table 4.2. Moderate or higher correlation scores are marked as bold.	78
4.5	Performance of The LSTM Models. Accuracy values are in percentage followed by Kappa values in brackets.	79
5.1	A sample question and answer between student and DeepTutor with the ideal expected answer.	83

5.2	Comparison of chunking accuracies of the various chunkers at chunk level (CL) and sentence level (SL) using gold chunks from the iSTS task 2015 data.	87
5.3	A sentence from Headlines training data chunked by our CRF chunker.	87
5.4	SemAligner output for a given text-pair.	89
5.5	F1 scores on gold and system chunked Headlines and Images training data of iSTS 2015 shared task.	90
5.6	F1 scores on gold and system chunked Images and Headlines test data. <i>MaxScore</i> is the best score for each metric given by any of the participating systems in the iSTS 2015 shared task.	91
6.1	A sample question and answer between student and DeepTutor with the ideal expected answer.	95
6.2	Results of different systems on CoNLL-2001 shared task test data. CM03 (Carreras & Marquez, 2004), CMPR02 (Carreras, Màrquez, Punyakanok, & Roth, 2002), CM01 (Carreras & Màrquez, 2001). P = Precision, R = recall, F= F-measure.	105
6.3	An optimal clause split generated from text: <i>the speed of the desk will increase since more force is being applied.</i>	105
6.4	A list of DT patterns for tuple extraction.	106
6.5	Summary of Data	111
6.6	An ordinal scale with four values for rating an extracted concept map of an ideal student answer along the metric <i>accuracy</i> .	112
6.7	Mean ratings for concept maps of ideal student answers generated by different open information extraction methods. The standard deviations are provided in bracket alongside means.	112
6.8	Results of different methods for binary classification. WA-Sim and WA-Sim-C are optimal word alignment-based STS system without context (Rus & Lintean, 2012) and with context (Banjade, Maharjan, Niraula, Gautam, et al., 2016) respectively. The value inside the bracket alongside the accuracy is Cohen's Kappa.	115
6.9	Performance of concept map based approach at the tuple level.	115

6.10 Performance on Multi-level classification task: Comparison of concept map approach against Logistic Model (Banjade, Maharjan, Niraula, Gautam, et al., 2016), GMM Model (Maharjan, Banjade, & Rus, 2017) and LSTM Model (Maharjan et al., 2018). The value inside the bracket alongside the accuracy is Cohen's Kappa.

116

Chapter 1

Introduction

1.1 Intelligent Tutoring Systems

Providing quality education to students requires sophisticated human tutors teaching them quality instructional materials. Because of the web, quality educational materials are easily accessible at affordable or no cost. However, the task of providing a quality human tutor service for each student anytime and anywhere at reduced costs is very challenging. Therefore, computer tutor applications called Intelligent Tutoring Systems (ITSs) that can mimic a human tutor can play an important role to achieve this goal.

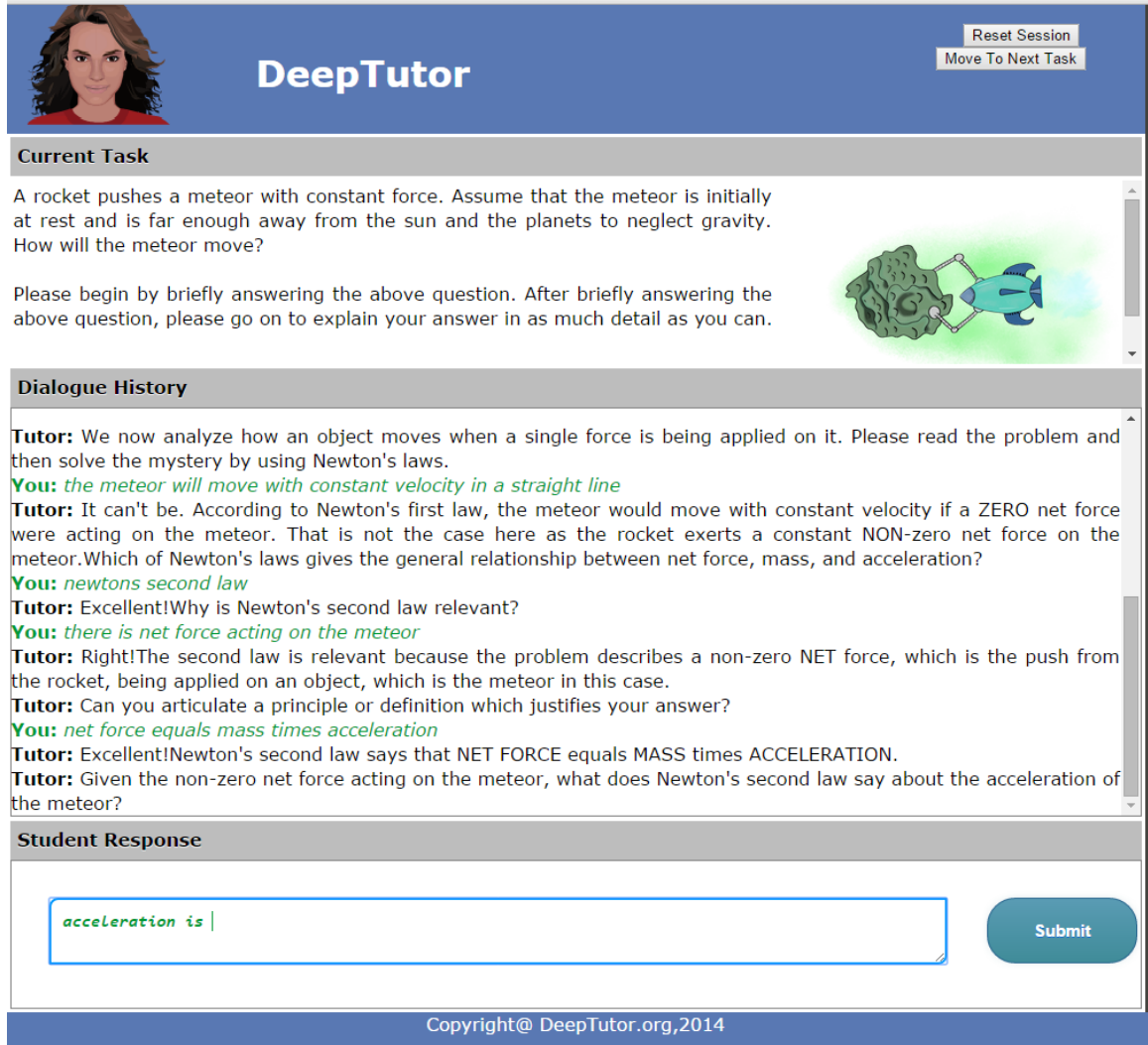
Intelligent Tutoring Systems (ITSs) are computer applications in the educational domain that can act as a personalized tutor in an interactive environment. They have been developed to teach students in many domains such as Science, Technology, Engineering and Maths (STEM) in a personalized and adaptive environment. Some instances of successful ITSs are AutoTutor (Graesser, Chipman, Haynes, & Olney, 2005), DeepTutor (Rus, DMello, Hu, & Graesser, 2013), Andes (Gertner & VanLehn, 2000), Crystal Island (Rowe et al., 2009), CircSimTutor (Evens et al., 1997), GuruTutor (Olney et al., 2012) and Why2 (VanLehn et al., 2007). Additionally, there is an ITS called iSTART (McNamara, Levinstein, & Boonthum, 2004) for reading comprehension and W-Pal for writing (McNamara et al., 2012).

VanLehn (2006) proposed the general architecture of an ITS consisting of two loops: *outer loop* and *inner loop*. The *outer loop* is responsible for serving learner-tailored content and tasks, i.e., the *outer loop* handles macro-adaptivity by managing the order of tasks (or problems) for the students to learn during tutoring. On the other hand, the *inner loop* provides a micro-adaptive environment to achieve

the goals of the current task undertaken by the student, where the student interacts with the tutor using natural dialogues or some on-screen elements. The *inner loop* typically consists of three stages: i) *the tutor asks a question*, ii) *the student provides the answer to the question* and then, iii) *the tutor assesses the answer*. If the student has mastered the task, the *inner loop* transfers the control to the *outer loop* to decide the next problem or task. Otherwise, the loop is repeated until the task is successfully learned.

Figure 1.1 an example of dialogue-based *inner loop* in the DeepTutor. The *Current Task* section describes the current problem or task attempted by the student along with related multimedia image. The *Dialogue History* section shows the history of the interactions between the student and the DeepTutor. The tutor starts the interaction by asking the student for a short-essay answer to a problem. The student types the answer in the *Student Response* section using natural English text. The tutor then assesses the student’s answer to check what expectations (solution steps) are covered and provides relevant feedback. If the student’s answer is incorrect with respect to an expectation, the tutor provides short feedback and provides hints and prompts to allow the students to make progress. In case the misconception is uncovered, the tutor explains to the student to correct his misconception. On the other hand, if the student response is correct, the tutor may provide positive feedback such as *Excellent!* and *Right!* Subsequently, the tutor asks the next question covering the next solution step. The task is completed once the student answers or covers all the expectations for the task.

Recent studies (VanLehn, 2011; Kulik & Fletcher, 2016) have shown that the Intelligent Tutoring Systems (ITSs) are as effective as average human tutors in inducing learning gains. Moreover, there is still the possibility that ITSs could be better than humans by further improving the micro-adaptivity functionality of the *inner loop*. In dialogue-based ITSs, effective dialogue and language processing



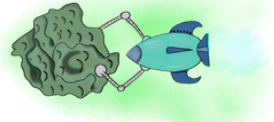
DeepTutor

Reset Session
Move To Next Task

Current Task

A rocket pushes a meteor with constant force. Assume that the meteor is initially at rest and is far enough away from the sun and the planets to neglect gravity. How will the meteor move?

Please begin by briefly answering the above question. After briefly answering the above question, please go on to explain your answer in as much detail as you can.



Dialogue History

Tutor: We now analyze how an object moves when a single force is being applied on it. Please read the problem and then solve the mystery by using Newton's laws.

You: *the meteor will move with constant velocity in a straight line*

Tutor: It can't be. According to Newton's first law, the meteor would move with constant velocity if a ZERO net force were acting on the meteor. That is not the case here as the rocket exerts a constant NON-zero net force on the meteor. Which of Newton's laws gives the general relationship between net force, mass, and acceleration?

You: *newtons second law*

Tutor: Excellent! Why is Newton's second law relevant?

You: *there is net force acting on the meteor*

Tutor: Right! The second law is relevant because the problem describes a non-zero NET force, which is the push from the rocket, being applied on an object, which is the meteor in this case.

Tutor: Can you articulate a principle or definition which justifies your answer?

You: *net force equals mass times acceleration*

Tutor: Excellent! Newton's second law says that NET FORCE equals MASS times ACCELERATION.

Tutor: Given the non-zero net force acting on the meteor, what does Newton's second law say about the acceleration of the meteor?

Student Response

acceleration is |

Submit

Copyright@ DeepTutor.org,2014

Fig. 1.1: Inner Loop Interface of DeepTutor tutoring system.

algorithms lie at the heart of the *inner loop*. In these systems, the *inner loop* relies on several components such as i) a dialogue based task script, ii) learner models, iii) models for handling dialogue acts and dialogue modes, iv) identifying and understanding effective tutorial strategies, v) assessment models for assessing the student answer and vi) a diagnostic feedback model. We focus on the last four components in our dissertation work.

1.1.1 Research Challenges

Many researchers have conducted research to identify effective tutorial strategies to use in the tutoring environment that induce learning gains (Aleven, Popescu, &

Koedinger, 2001; Cade, Copeland, Person, & DMello, 2008). However, the task of identifying effective tutorial strategies is very challenging given that average human tutors rarely employ sophisticated tutoring (Graesser, DMello, & Person, 2009). Therefore, there is a need to discover and understand tutoring strategies used by expert tutors, as opposed to average tutors, or are motivated by pedagogical theory. The latter approach is to implement pedagogically backed strategies in an ITS and validate them through controlled experiment (Rus, DMello, et al., 2013). The other approach is to mine the existing huge amount of tutorial data to discover effective tutorial strategies. The effective strategies, once discovered, can be implemented in ITSs that could re-enact these strategies systematically thus leading to effective ITSs which in turn would lead to an improved educational ecosystem.

Another challenge is to correctly evaluate student answers. The true understanding of the student answer is an intractable problem since it requires making inferences over linguistic knowledge, world knowledge, domain knowledge, and contextual information. Therefore, the widely adopted and scalable approach to assessing such open-ended student responses is the semantic similarity in which a score is computed between a target student answer and an expert-provided reference answer. The underlying assumption, in this case, is that the students provide explicit and self-contained answers. However, the student answers are often short involving anaphora and ellipsis. This phenomenon is more common in an interactive conversational tutoring environment such as DeepTutor. For example, Niraula et al. (2014) analyzed tutorial conversational logs and showed that 68% of the pronouns used by students were referring to entities in the previous dialogue turn or in the problem description.

Additionally, they found that about 65% of pronouns in the student answers have their referents in the immediate previous utterance, i.e. tutor question and the problem description. Based on this finding, Banjade and colleagues (Banjade,

Table 1.1: A sample question and answer between student and DeepTutor with the ideal expected answer.

Description
Question: Because it is a vector, acceleration provides what two types of information?
Student Answer: Acceleration gives magnitude
Expected Answer: Acceleration provides magnitude and direction.

Maharjan, Niraula, Gautam, et al., 2016) used as context the previous utterance and the problem description while annotating 900 student responses collected from the logged interactions of 40 different students in an experiment with the DeepTutor (Rus, DMello, et al., 2013) to create the DTGrade dataset. On analyzing DTGrade dataset, the authors showed that 25% of the annotated instances require context to assess the correctness of the student answers. This highlights the vital role of context when assessing student answers particularly, in conversational tutoring environments.

The other issue is interpreting the semantic similarity score of a student answer. The Semantic Textual Similarity (STS) systems merely measure the degree of similarity, i.e., the correctness of the student responses when compared against the expert provided answers. However, these systems don't explain why the two texts are equivalent, or similar or unrelated. They do not give any information about which parts of the sentences are equivalent in meaning (or very close in meaning) and which not. For example, consider a question asked by the DeepTutor, its expected answer and one of the student responses to the question as shown in the Table 1.1.

A typical STS system does not indicate which information is missing in the student answer for the above question. If there existed explanatory functionality that could explain the missing information is *direction*, we could use this diagnostic information to generate a follow-up question by the tutor as *What other information*

than magnitude is provided by the acceleration? Therefore, the capability to explain the similarity in addition to similarity score is important in applications such as intelligent tutoring systems because this empowers the *inner loop* to improve the micro-adaptivity by providing relevant diagnostic feedback to the student.

Our work is motivated by our interest to improve the overall learning experience of the students with the conversational intelligent tutoring systems by seeking solutions to the challenges described above.

1.2 Goal

Our work is aimed at improving online Intelligent Tutoring Systems. Towards meeting this goal, we describe various methods and approaches to improving the assessment of student responses (with and without contextual information) and, providing interpretable textual similarity capability for giving appropriate and relevant feedback. We also describe our approach to identify and understand effective pedagogical strategies employed by successful tutors that lead to learning gains.

1.3 Research Questions

The key research questions we target to answer in this research work are outlined below.

- How to identify and understand effective tutorial strategies employed by successful tutors that yield tutoring sessions with learning gains?
- How to improve generic sentence similarity methods for assessing short texts?
- How to improve automated assessment of open-ended student answers in tutorial dialogue using contextual information?
- How to interpret the predicted similarity score and provide subsequent diagnostic feedback in tutorial dialogue contexts?

- How to automatically extract student mental model representations from tutorial dialogues in the form of entity-relations graphs or conceptual maps and then use them for assessing student performance?

1.4 Summary of Primary Contributions

We describe the contributions of our dissertation work below.

In Chapter 2, we describe our supervised approach to map tutor-tutee utterances in tutorial sessions onto actions by classifying them into dialogue acts, dialogue sub-acts and dialogue modes. We then analyze sequences of tutor and tutee actions to identify the most interesting and useful patterns associated with effective tutorial sessions, i.e., sessions in which there is evidence of tutee learning gains.

In Chapter 3, we present our various methods to measure the similarity between short sentences including our novel approach based on Gaussian Mixture Model to measure short textual similarity. We also describe our different Support Vector Regression (SVR) models which were top performing systems for Semantic Textual Similarity in Semantic Evaluation (SemEval) in different years. SemEval is an ongoing series of evaluations of computational semantic analysis systems. Our initial SVR models (Banjade, Niraula, et al., 2015; Banjade, Maharjan, Gautam, & Rus, 2016) used multiple features including multi-level alignment and vector-based compositional semantic similarity measures. In addition to the above features, our recent SVR model (Maharjan, Banjade, Gautam, Tamang, & Rus, 2017) includes sentence-level embeddings and Gaussian Mixture Model similarity.

In Chapter 4, we present our approach to assessing open-ended student answers in context. Evaluating student answers in context is particularly important in an Intelligent Tutoring System like DeepTutor (Maharjan, Banjade, & Rus, 2017), where student answers vary significantly in their explicit content and writing style. We describe our probabilistic Gaussian Mixture Models (GMMs) and Long

Short Term Memory (LSTM) models to assess student answers which takes context into account.

In Chapter 5, we present our SemAligner tool which can be useful for explaining or interpreting the similarity score between two texts by identifying chunks and aligning them across the two texts indicating the semantic relation and similarity score of each alignment. We use chunk alignment types :- *EQUI* (semantically equivalent), *OPPO* (opposite in meaning), *SPE* (one chunk is more specific than other), *SIMI* (similar meanings, but not *EQUI*, *OPPO*, *SPE*), *REL* (related meanings, but not *SIMI*, *EQUI*, *OPPO*, *SPE*), and *NOALI* (has no corresponding chunk in the other sentence). The relatedness/similarity scores are assigned in the range of 0 to 5.

Next, we describe our novel concept map-based method in Chapter 6 that can both assess and interpret the student answers in tutorial dialogues. In this approach, we automatically extract concepts and relations from the student responses to build a concept map for each student for a given problem. The concept map would serve as a representation of the mental model of the student understanding of the problem and target domain. Once we have the concept map for the student, we assess the student knowledge by comparing the student concept map with the ideal concept map for the problem created by Subject Matter Experts (SMEs).

We use concept maps for both evaluating and explaining the degree of correctness of the student answers. We derive a concept map for the current student answer and compare it against the corresponding concept map for the ideal answer to determine the degree of correctness. For example, we can find out that the student answer is missing the *direction* concept from the expected answer for the question asked in Table 1.1 and therefore, we can provide appropriate feedback and plan the next dialogue moves for the missing concept.

Chapter 2

Discovering Effective Tutorial Strategies Employed by Professional Tutors

One of the key research questions in the tutoring community is: *What effective tutorial strategies are employed by expert tutors that induce learning gains?* The task is very challenging, as average human tutors rarely employ sophisticated tutoring strategies (Graesser et al., 2009). Therefore, there is a need to discover and understand tutoring strategies that are either manifested by expert tutors, as opposed to average human tutors, or are motivated by pedagogical theory.

In the pedagogical theory approach (Aleven et al., 2001; Rus, Banjade, Niraula, Gire, & Franceschetti, 2017), sound pedagogical strategies based on theory are typically implemented in an intelligent tutoring system (Rus, DMello, et al., 2013) and are validated through controlled experiments. The other is a data-driven approach, which we adopt here, to discover strategies from expert tutors by mining existing tutoring data collected from online human tutoring services (Boyer et al., 2011; Cade et al., 2008; Rus, Maharjan, & Banjade, 2015; Ohlsson et al., 2007).

However, understanding the effective strategies by studying the strategies used by the expert tutors is a hard problem because what characterizes tutoring expertise is an open question (Rus, Maharjan, & Banjade, 2015). A tutor who employs sound strategies may appear less expert when working with students having low abilities or lacking in motivation. On the other hand, an average tutor may seem expert if he only works with highly able and motivated students. Also, Ohlsson and colleagues (Ohlsson et al., 2007) reported in their study that the number of years of tutoring experience and pay scale, which are typically used as proxies for expertise, of the tutors do not impact the average learning gains. It should be noted that Ohlsson and colleagues used a small number of tutors in their study.

Consequently, we distinguish in our work between *effective* tutoring (the kind that induces learning gains) and *expert* tutoring (do the right thing, e.g., following sound pedagogical standards). It should be noted that this distinction is similar to research work in teacher expertise (Berliner, 2001) that distinguishes between good versus successful teachers: good teachers are those whose classroom performance meets professional teaching standards, whereas successful teachers are those whose students achieve set learning goals. It is beyond the scope of this paper to fully address the topic of tutoring expertise. Instead, we focus on effective tutoring by identifying effective tutors which in turn are identified by identifying effective tutorial sessions.

Once effective sessions are identified, we need to characterize and explore tutors' actions. For this, we map the dialogue-based interactions, which are streams of utterances, into streams of actions (dialogue acts) based on the language-as-action theory (Austin, 1975; Searle, 1969) (described in Section 2.2) using a predefined dialogue taxonomy (described in Section 2.3). Once tutorial sessions were mapped onto sequences of dialogue acts and dialogue modes, we attempted to identify patterns of actions that are associated with learning gains. We inferred different patterns over the tutor-tutee actions sequences through tutorial session analysis (Rus, Maharjan, Lasang, et al., 2017). We describe our approach for identifying patterns in effective and ineffective tutorial sessions in Section 2.4. We analyzed what do good human tutors do in such good sessions. We also compared the profiles (distributions of dialogue acts/dialogue modes) for effective versus less effective sessions to understand and characterize effective human tutoring. Moreover, we also investigated what effective strategies, i.e. sequences of tutor actions, can be discovered in good tutoring sessions but not in bad tutoring sessions. In this work, we also report our findings with respect to dialogue-act, sub-act and dialogue-mode classification (see Section 2.6.2).

2.1 Related Work

Discovering the structure of tutorial dialogues and tutors’ strategies has been one of the main goals of the intelligent tutoring research community for quite some time. For instance, Graesser, Person, and Magliano (1995) explored collaborative dialogue patterns in tutorial interactions and proposed a five-step general structure of collaborative problem-solving during tutoring.

Over the last decade, the problem has been better formalized and also investigated more systematically using more rigorous analysis methods (Cade et al., 2008; Boyer et al., 2011). For example, tutoring sessions are segmented into individual tutor and tutee actions and statistical analysis and artificial intelligence methods are used to infer patterns over the tutor-tutees action sequences. The patterns are interpreted as tutorial strategies or tactics which can offer both insights into what tutors and students do and guidance on how to develop more effective intelligent tutors that implement these strategies automatically. Our work contributes to this area of research by exploring tutors’ actions by doing a tutorial data analysis at scale, i.e., using a big collection of tutorial data.

2.2 Language as Action

The actions of speakers can be represented by dialogue acts inspired from the language-as-action theory (Austin, 1975; Searle, 1969) which states that *when we say something we do something*.

An utterance can be thought of as serving an action. For example, a simple utterance of *‘Hello!’* represents an expression of a greeting action. *‘Could you please pass me the book?’* serves a request action. A dialogue act may have some finer subtleties, and at the lower level, the utterance can be labeled with a dialogue act and sub-act combination. For example, the tutor utterance (T1) in the Table 2.1: *“There is an useful idea called ‘conservation of energy’”* can be categorized as an *Assertion* dialogue act, i.e. the utterance is making an assertion about the

Table 2.1: A snippet of tutorial dialogue between student and tutor labeled with dialogue acts and modes. PK refers to Prior Knowledge.

Speaker	Act	Sub Act	Mode	Utterance
Tutor	<i>Assertion</i>	<i>Concept</i>	<i>Telling</i>	There is an useful idea called 'conservation of energy'
Student	<i>Assertion</i>	<i>PK: Positive</i>	<i>Telling</i>	Yes. I know about that too.

“*conservation of energy*” which is a *Concept* sub-act. Similarly, the subsequent student utterance (S1): “*Yes. I know about that too.*” represents an *Assertion* dialogue act initiated by the student having some prior knowledge on the concept. Therefore, it has sub-act label ‘*PriorKnowledge: Positive*’. Similarly, all the dialogue utterances in a dialogue interaction can be labeled with corresponding dialogue acts and sub-acts.

We then further group these dialogue acts and sub acts into higher concepts called dialogue modes. For example, the utterances T1 (*Assertion: Concept*) and S1 (*Assertion:Prior Knowledge:Positive*) are part of chunks of actions related with the dialogue mode *Telling*. Therefore, a dialogue mode serves a particular task-related or pedagogical goal. To conclude, we map tutorial sessions consisting of streams of utterances into streams of dialogue acts and dialogue modes.

2.3 Dialogue Taxonomy

The current coding taxonomy is adapted to our context from an earlier taxonomy built over a large corpus of online tutoring sessions conducted by human tutors in the domains of Algebra and Physics (Morrison et al., 2014). It should be noted that the dialogue acts and subacts were defined and refined to minimize overlap between categories and maximize the coverage of distinct acts. It is more granular than previous schemes such as the one used by Boyer and colleagues (Boyer et al., 2011). There are 17 top level expert-defined dialogue act categories: *Answer*, *Assertion*, *Clarification*, *Confirmation*, *Correction*, *Directive*, *Explanation*, *Expressive*, *Hint*, *LineCheck*, *Offer*, *Promise*, *Prompt*, *Question*, *Reminder*, *Request* and *Suggestion*.

The *Prompt* and *Hint* are two additional pedagogical categories included in the current taxonomy. Each dialogue act category may have 4 to 22 subcategories or sub-acts. For example, we distinguish *Assertions* that reference aspects of the tutorial process itself (*Assertion:Process*); domain concepts (*Assertion:Concept*), or the use of lower-level mathematical calculations (*Assertion:Calculation*). The taxonomy identifies 129 distinct dialogue act and sub-act combinations. Further, we have a set of 17 different dialogue modes defined by the experts as: *Assessment*, *Closing*, *Fading*, *ITSupport*, *Metacognition*, *MethodID*, *Modeling*, *OffTopic*, *Opening*, *ProblemID*, *ProcessNegotiation*, *RapportBuilding*, *RoadMap*, *SenseMaking*, *Scaffolding*, *SessionSummary* and *Telling*.

The set of 17 expert-defined modes used by the SMEs to manually label the data is shown below. A detailed description of the dialogue modes is available (Morrison et al., 2014).

- *Assessment*: A mode in which the tutor “assesses the student’s level of understanding to determine a suitable starting point.”
- *Closing*: A mode in which the tutor moves to close out the session, often ending in an exchange of farewells, and the tutor’s reminder to complete a satisfaction survey.
- *Fading*: A mode in which the student is working on the problem successfully, with only occasional contributions from the tutor, such as in the form of positive confirmation or praise.
- *ITSupport*: A mode in which the conversation is about IT related issues.
- *Metacognition*: A mode in which the tutor makes assertions to help the student think about the problem-solving process at a “meta” level, e.g., the importance of perseverance, checking for careless errors, etc.

- *MethodID*: A mode during which the tutor attempts to determine if there is a particular method the student needs to use to solve the problem.
- *Modeling*: A mode in which the tutor is showing the student how to solve a problem, such as by completing one or more of the steps herself.
- *Off Topic*: A mode in which the students engages the tutor in an off-topic conversation.
- *Opening*: The opening mode of the session (e.g., greetings, asking for help).
- *ProblemID*: A mode in which the tutor seeks to “identify and achieve a clear understanding of the student’s needs/expectations.”
- *Process Negotiation*: A mode in which the two conversational partners negotiate aspects of the tutorial process itself such as whether there is time to work on another problem, etc.
- *Rapport Building*: A mode in which the tutor or tutee or both intend primarily to build rapport, e.g., expressions of praise, apologies, affect queries (How are you this evening?).
- *RoadMap*: A mode during which the tutor lays out the “game plan” for solving the problem.
- *Sensemaking*: A mode in which the tutor is helping the student gain conceptual understanding as opposed to procedural accuracy, e.g., explaining why a particular problem-solving step is appropriate in a particular circumstance.
- *Session Summary*: A mode in which the tutor reviews the session, with an emphasis on what the student might have or ought to have learned.

- Scaffolding: A mode in which the student is doing most of the work and the tutor making frequent contributions of assistance.
- *Telling*: A mode in which the tutor asserts and explains, with relatively few student contributions mostly in the form of confirmations of understanding.

2.4 Identifying Patterns in Effective Tutorial Sessions

We identify effective tutorial sessions by using both expert human judgments and objective learning gain measures. Our research corpus consists of dialogue-based tutorial sessions involving interactions between professional tutors and learners from an online provider of human tutoring services. There are no pre- and post-tests for these chat-based tutorial sessions to infer learning gains. For annotated corpus, the annotators rated the annotated tutorial sessions with the scores for evidence of learning (EL) and evidence of soundness (ES). We use an average of EL and ES to measure the quality of tutorial sessions. We also use pre- and post-tutoring measures of mastery of target topics to derive the learning gains by aligning the human tutoring data with sessions offered by Carnegie Learning’s Cognitive Tutor (Ritter, Anderson, Koedinger, & Corbett, 2007). Students in our sessions are college-level, adult students who are required to interact with Cognitive Tutor and also have the option to ask for help from a human tutor. It is important to note that most students do not ask for help from a human tutor (Ritter, Fancsali, Yudelson, Rus, & Berman, 2016) which may imply a self-selection bias in our student population in the sense that it might consist of students that have higher meta-cognitive skills, e.g., they self-assess their knowledge and affective states and decide to ask for more help if needed, or prefer social interactions or appreciate affective support from a knowledgeable other human being. The results we present here should be interpreted with this important aspect of our data in mind.

Once effective sessions are identified, we need to characterize and explore tutors’ actions. For this, we map the dialogue-based interactions, which are streams

of utterances, into streams of actions (dialogue acts) based on the language-as-action theory (Austin, 1975; Searle, 1969) (described in Section 2.2) using a predefined dialogue taxonomy (described in Section 2.3). We adopted a supervised machine learning method to automate the mapping of each utterance into a dialogue act and sub-act label (Rus, Maharjan, & Banjade, 2017; Rus et al., 2016). Further, we applied a supervised approach to labeling dialogue modes using a sequence labeling framework based on Conditional Random Fields (Rus, Niraula, Maharjan, & Banjade, 2015). A dialogue mode is a chunk of actions associated with general conversational segments and is related to a task or pedagogical goal. For instance, during a learner-tutor interaction, there might be a segment of interactions where the tutor would be exemplifying and explaining (modeling) the application of certain concepts. We call such as segment *Modeling* mode.

Once tutorial sessions were mapped onto sequences of dialogue acts and dialogue modes, we attempted to identify patterns of actions that are associated with learning gains. We inferred different patterns over the tutor-tutee actions sequences through tutorial session analysis (Rus, Maharjan, Lasang, et al., 2017). As Rus and colleagues (Rus, Conley, & Graesser, 2014) noted, there could be only one tutoring strategy which is to make the learner apply effective learning strategies which, in turn, implies that we need to also analyze what tutees do in response to tutors' actions. We analyzed and explored different patterns in terms of dialogue acts and modes to understand human tutoring sessions. We investigated the typical dialogue act and mode profiles of an effective tutoring session. We also analyzed what do good human tutors do in such good sessions. We also compared these profiles for effective versus less effective sessions to understand and characterize effective human tutoring. Moreover, we also investigated to see what effective strategies, i.e. sequences of tutor actions can be discovered in good tutoring sessions but not in bad tutoring sessions. These patterns or tutorial strategies can offer

insights into what strategies tutors and learners employ and therefore offer guidance on how to develop more effective intelligent tutoring systems (ITSs). For instance, strategies used by effective human tutors, once discovered, can be implemented in ITSs that could re-enact these strategies systematically thus leading to effective ITSs which in turn would lead to an improved educational ecosystem.

Next, we describe a set of different features used to develop classifiers for automatically tagging utterances in tutorial sessions for dialogue acts, sub-acts and modes.

2.5 Feature Selection

Representing an utterance by a set of features is a non-trivial task. The standard approach for selecting features is to consider a rich set of features including n-grams, POS and lemma and, then applying some feature selection methods to avoid over-fitting. However, firstly it is not clear whether there is a need for so many features to solve the dialogue classification problem. Second, it is difficult to interpret how individual features contribute to a classification task. Therefore, we use few intuitive and meaningful features to represent a dialogue utterance in our supervised approach to classifying dialogue utterances as dialogue act, sub-act, and mode labels. Our experiments are meant to validate our selection of features.

We selected these features based on the assumption that humans infer speakers' intention after hearing only a few of the leading words of an utterance (Moldovan, Rus, & Graesser, 2011). One argument in favor of this assumption is the evidence that hearers start responding immediately (within milliseconds) or sometimes before speakers finish their utterances (Martin & Jurafsky, 2000).

Intuitively, the first few words of a dialogue utterance are very informative of that utterance's dialogue act. For instance, *Questions* usually begin with a *Wh*-word while dialogue acts such as *Answers* contain a semantic equivalent of yes or no among the first words, and *Greetings* use a relatively small bag of words and

Id	Feature Description
F1	first token
F2	second token
F3	third token
F4	last token
F5	utterance length
F6	dialogue act
F7	dialogue act - subact
F8	bigrams of F1-F2 and F2-F3
F9	trigram of F1-F2-F3
F10	{F1, F2, F3, F4} of last 2 utterances
F11	{F1, F2, F3, F4} of next 2 utterances
F12	F6 of last utterance
F13	F6s of last 2 and next 2 utterances
F14	F7 of last utterance
F15	F7 of last 2 and next 2 utterances
F16	dialogue mode of last utterance

Table 2.2: Features

expressions. In the case of other dialogue act categories, distinguishing the dialogue act after just the first few words is not trivial, but possible. It should be noted that in typed dialogue, which is less expressive than spoken dialogue, some information, such as intonation is lost. We should also recognize that the indicators allowing humans to classify dialogue acts also include the expectations created by previous dialogue acts. For instance, after a first greeting, another greeting, that replies to the first one, is more likely. Therefore, our primary feature set consists of *first token*, *second token*, *third token*, *last token* and *utterance length* to represent a dialogue utterance.

We summarize all the features used in our experiments in Table 2.2.

Dialogue act is the first step towards classifying utterances in a tutorial session. So, we used utterance level primary features (F1-5) and their contextual derivatives such as bigrams of F1-F2 and F2-F3 (features F7 and F8), trigram of F1-F2-F3 (feature F9), {F1, F2, F3, F4} of previous two and next two utterances (features F10 and F11) for dialogue act classification. For next dialogue act - subact

Human Transcript Annotation Software

The screenshot displays a table for transcript annotation with columns: Timestamp, Speaker, Utterance, Tag, Dialog Act, Subtype, Dialog Mode, and Comments. A blue arrow points to the 'Tag' column. A modal window titled 'Tagging Data' is open over the second row.

Timestamp	Speaker	Utterance	Tag	Dialog Act	Subtype	Dialog Mode	Comments
00:00:08	Tutor	Hi		Expressive	Greeting	Opening	
00:00:21	Student	hello jim					
00:00:45	Tutor	Do you know what slope-intercept form looks like?		Request	Confirmation:PriorKnowledge	Assessment	
00:01:00	Student	no		Confirmation	Negative		

Tagging Data

Dialog Act:

Dialog Sub-Act:

Dialog Mode:

☐ I am NOT sure that this is correct (ALT+C)

Act: Expressive, SubAct: Greeting, Mode:

Comments:

Fig. 2.1: Tutorial Data Annotation.

classification, we used additional richer features at dialogue act levels such as current dialogue act (F6) and dialogue acts of the last two and next two utterances (F13). Finally, our CRF based dialogue mode labeler used even richer features at dialogue act and dialogue act-subact levels, e.g. F12, F13, F14, F15 in Table 2.2.

It should be noted that we experimented with other features such as the speaker (student vs. tutor), the position of the utterance in the dialogue, e.g., an utterance at the beginning of a session is more likely a greeting, the previous dialogue act. However, these features didn't improve model performance.

2.6 Experiments and Results

2.6.1 Data

A large corpus of about 19K tutorial sessions between professional human tutors and actual college-level, adult students was collected via an online human tutoring

service. Students taking two college-level developmental mathematics courses (pre-Algebra and Algebra) were offered these online human tutoring services at no cost. The same students had access to computer-based tutoring sessions through Adaptive Math Practice, a variant of Carnegie Learning’ Cognitive Tutor. A subset of 500 tutorial sessions containing 31,299 utterances was randomly selected from this large corpus for annotation with the requirement that a quarter of these 500 sessions would be from students who enrolled in one of the Algebra courses (Math 208), another quarter from the other course (Math 209), and half of the sessions would involve students who attended both courses.

Data Annotation Process: Figure 2.1 shows the process of annotating the randomly selected 500 tutorial sessions. The sessions were manually labeled by a team of 6 subject matter experts (SMEs), e.g., teachers that teach the target topics. They were trained on the taxonomy of dialogue acts, sub-acts, and modes. Each session was manually tagged by two independent annotators without looking at each other’s tags to eliminate the labeling bias problem using web-based transcript annotation software. The tags of the two independent annotators were double-checked by a verifier who resolved discrepancies in tags if any. The verifier also happens to be the designer of our dialogue taxonomy. The average inter-annotator agreement for the two independent annotators is summarized in Table 2.3.

Based on the assumption that tutors often switch from one particular instructional strategy (dialogue mode) to another strategy during tutoring, the annotators labeled only the utterances where a switch in strategy occurred. For instance, they annotated an utterance with the dialogue mode label of *ProblemIdentification* where a switch occurred from, say, an *Opening* to the *ProblemIdentification* mode. So, it should be noted that the dialogue mode

Annotation	Agreement (%)	Kappa
Act	77	0.72
Act-subact	62	0.60
Mode	44	0.37
Mode*	53.8	0.48
Mode**	64.3	0.60

Table 2.3: Average Inter Annotator Agreement Between Two Independent Annotators. Mode* and Mode** represent dialogue mode agreement between verifier and first annotator and, verifier and second annotator respectively.

agreement above refers to mode-switches. In our context, dialogue mode should be interpreted as dialogue mode-switch throughout the paper.

When we looked at the confusion matrix, the most confusing modes for annotators were *Fading*, *Scaffolding*, *ProcessNegotiation* and *ProblemIdentification*. They confused *Scaffolding* with *Fading* 16%, *Fading* with *Scaffolding* 28%, *ProblemIdentification* with *ProcessNegotiation* 17% and *ProcessNegotiation* with *ProblemIdentification* 15% of the time. However, the tagging discrepancies were resolved by the verifier. The mode agreement between verifier and first annotator and verifier and second annotator were (53.8%, kappa = 0.48) and (64.3%, kappa = 0.60).

2.6.2 Dialogue Classification

We built different classifiers based on a supervised machine learning approach for predicting the dialogue act, sub-act and dialogue mode labels. We used the above 500 annotated tutorial transcripts for training our classifiers and evaluated their performance in terms of accuracy and Cohen’s kappa relative to the final tag adjudicated by the verifier using a 10-fold cross-validation approach.

Dialogue Act Classification: For dialogue act classification, we describe two of our best classifiers. We trained a Hidden Markov Model (HMM) in which we assumed the dialogue acts as the hidden states generating the observations, i.e. dialogue utterances in temporal space. We represented the dialogue utterance by

Classifier	Accuracy (%)	Kappa
HMM	67.9	0.59
CRF	74.3	0.67

Table 2.4: Performance of dialogue act classifiers

Classifier	Accuracy (%)	Kappa
CRF	51.2	0.49
Liblinear	53.47	0.67
CRF-2*	65	0.63

Table 2.5: Performance of dialogue act - subact classifiers. CRF-2 uses gold dialogue acts to analyze impact of noisy predicted acts.

the set of F1-5 features in Table 2.2. Finally, we used the trained HMM to predict dialog act labels by finding the most probable sequence of dialogue acts (hidden states) given a sequence of utterances in a tutorial session. In our second approach, we used discriminative Conditional Random Fields (CRF) to tackle the dialogue act classification as a sequence labeling task. CRFs do not suffer from the label bias problem like Maximum Entropy Markov Models (MEMMs). Also, unlike HMM, the CRF model may account for the full context of a set of observations using features of various levels of granularity. We used F1-5 and F8-12 as features for our CRF based dialogue act classifier.

Dialogue Act-Subact Classification: We predicted dialogue act-subact labels in two steps. We first used a CRF-based dialog act classifier (described above) to predict dialogue acts for given utterances. Then, we used CRF subact classifier to predict the subsequent subact label using utterance features and predicted dialogue acts. Our subact classifier used F6, F13 and F14 in addition to F1-5, F8-11 as features. Due to cascaded design, our dialogue act-subact CRF model performance may suffer in the case predicted dialogue act is incorrect. Next, we built a flat dialogue act-subact classifier using a liblinear kernel with logistic loss function using F1-5 as features.

Dialog Mode Labeling: We labeled dialogue modes by considering it as

Classifier	Accuracy (%)	Kappa
CRF	51.7	0.48

Table 2.6: Performance of dialogue mode classifier

both classification and sequence labeling tasks using CRFs. In this case, we used current dialogue act (F6), current act-subact (F7), dialogue acts and act-subacts of previous two and next two utterances (F13, F15) and dialogue mode of the previous utterance (F16) as features to build our CRF model.

Results: Table 2.4 shows the performance of our dialogue act classifiers. The CRF based dialogue act classifier outperforms its HMM counterpart and yields the best accuracy of 74.3% with a kappa of 0.67. Also, the performance of our CRF classifier is close to the inter-rater agreement for the dialogue acts (see Table 2.3). For dialogue act-subact classification, the Liblinear model slightly outperforms the CRF classifier as shown in Table 2.5 but it is not close to inter-annotator agreement value. The CRF classifier might have suffered due to noisy predicted dialogue acts. We validated this assumption by building another CRF model (CRF2 in Table 2.5) where we provided true dialogue act values for features F6 and F13. The CRF2 model resulted in 65% accuracy with a kappa of 0.63 which is even better than the inter-rater agreement for dialogue act - subact labeling. Our dialogue mode classifier yielded an accuracy of 51.7% and kappa=0.48. Interestingly, the accuracy and kappa for the dialogue mode labeling are better than the human inter-annotator agreement (see Table 2.3). This is explained by the fact that our learning method captures well the final labels assigned by the final verifier and therefore agrees more with the final labels than the two independent annotators.

2.6.3 Tutorial Session Analysis

We performed a number of analyses of tutorial sessions to understand the general structure of such sessions and identifying patterns of actions that are linked to

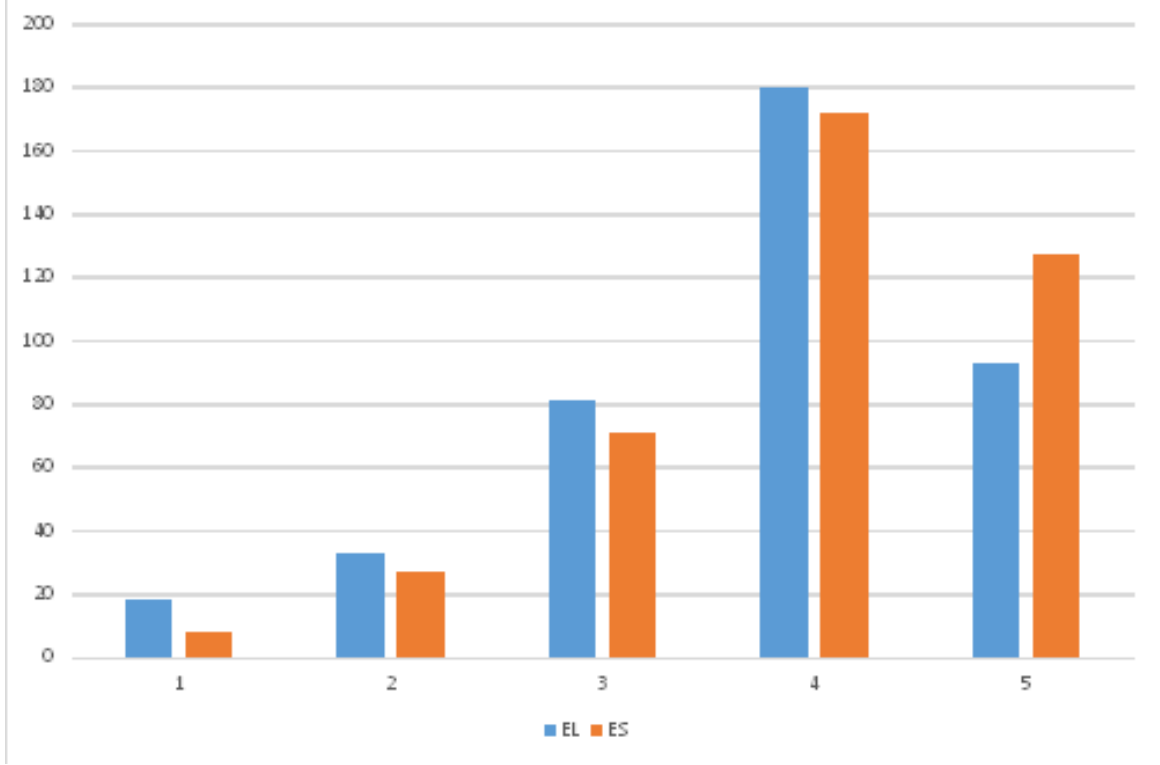


Fig. 2.2: Distribution of human ratings of Evidence of Learning (EL) and Evidence of Soundness (ES).

learning gains. We used two measures to represent the learning gain: a *human judgment measure* and a *computer-generated measure*.

2.6.4 Analysis using human judgment measure of learning gain

We first used human judgment measures to identify effective and ineffective sessions. The SMEs rated each tutorial session using a 1-5 scale (5 being the highest/best score) along the two dimensions of evidence of learning (EL) and evidence of soundness (ES). The EL and ES were found to be highly correlated with a Pearson coefficient of 0.7. However, 55 sessions annotated in the training phase of the annotation did not have these ratings. In addition to this, 40 out of 445 annotated sessions did not have rating information of EL. Figure 2.2 shows the distribution of the ratings of EL and ES given by the annotators. We see that most annotated sessions are dominantly good sessions having both EL and ES rated ≥ 4 . On the other hand, about 12% sessions were bad sessions (EL and ES rated ≤ 2).

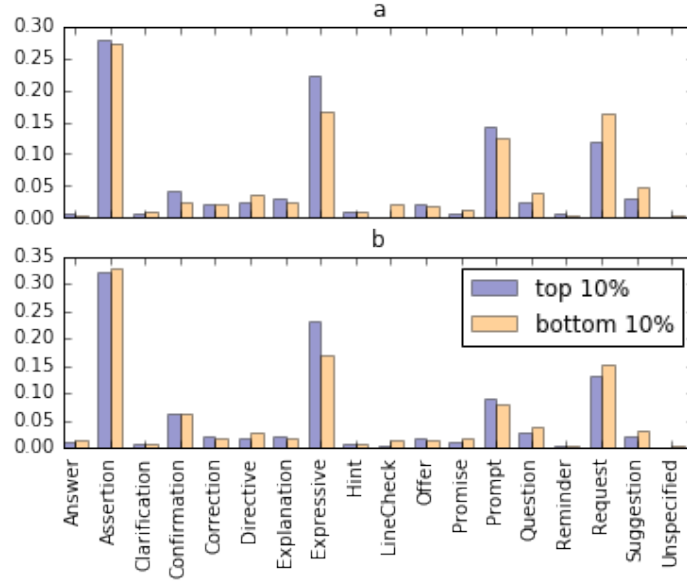


Fig. 2.3: Distribution of dialogue act profile of top versus bottom 10% sessions. a show the profile for tutors only while b shows the profile for both tutors and students.

We used both EL and ES measures to identify effective and ineffective sessions. There might be sessions in which students might not learn much despite the best application of pedagogically sound tactics. On the other hand, a high-ability student might learn even if the tutor is not applying accepted pedagogically sound strategies. Therefore, we consider an average of the learning and soundness scores to capture these different situations and generate a final score the captures the overall quality of the tutorial sessions.

Profile comparison analysis: As a first analysis, we conducted a comparison of the distributions of dialogue acts for top 10% sessions versus bottom 10% sessions when ranked based on the holistic score, i.e., the average EL and ES ratings. We compared distributions of dialogue acts (dialogue act profiles) for the tutors only and for both tutors and students (Figure 2.3 *a* and *b*), respectively.

We found that tutors in good sessions generate, on average, more *Expressives* (22.2% vs 16.5%, p-value < 0.001) and *Prompts* (14.3% vs 12.3%, p-value = 0.15) and less *Requests* (11.8% vs 16.3%, p-value = 0.002) than those in the bottom 10%

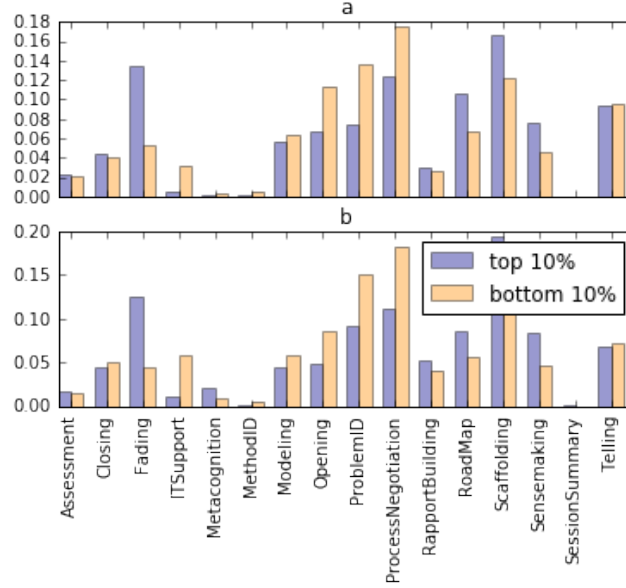


Fig. 2.4: Distribution of mode switch profile of top versus bottom 10% sessions. a show the profile for tutors only while b shows the profile for both tutors and students.

sessions. Even when considering the dialogue act profile for students and tutors together, there are relatively more *Expressives* (23.3% vs 16.7%, p-value < 0.001) and less *Requests* (12.9% vs 15.1%, p-value = 0.051) acts in the top 10% sessions than in the bottom 10% sessions.

Similarly, we investigated the dialogue mode switch profiles for top 10% good sessions against bottom 10% bad sessions as shown in Figure 2.4. The mode profile revealed that there are relatively more *Fading* (13.4% vs 5.2%, p-value < 0.001), *Scaffolding* (16.6% vs 12.2%, p-value = 0.066) and *RoadMap* (10.5% vs 6.6%, p-value = 0.047) modes initiated, on average, by the tutors in the top 10% sessions. On the other hand, there are relatively less *ProcessNegotiation* (12.3% vs 17.4%, p-value = 0.028), *ITSupport* (0.5% vs 3.2%, p-value < 0.001), and *ProblemID* (7.3% vs 13.6%, p-value = 0.001) modes initiated, on average, by tutors in the good sessions. These observations are true even when we look at the mode switch profile across both tutors and students (figure not shown due to space limitations) i.e. there are relatively more *Scaffolding* (19.3% vs 12.6%, p-value = 0.002), *Fading*

Dialogue Mode	Symbol
Assessment	A
Closing	C
Fading	F
ITSupport	I
Metacognition	M
MethodID	E
Modeling	D
Opening	O
ProblemID	P
ProcessNegotiation	N
RapportBuilding	B
RoadMap	R
Scaffolding	S
SenseMaking	K
SessionSummary	Y
Telling	T

Table 2.7: Mapping of dialogue modes to symbols.

(12.6% vs 4.3%, p-value = 0.001) and *RoadMap* (8.6% vs 5.6%, p-value = 0.056) and less *ProcessNegotiation* (11.1% vs 18.3%, p-value < 0.001), *ITSupport* (1.1% vs 5.8%, p-value < 0.001) and *ProblemID* (9.2% vs 15.0%, p-value = 0.002) in the top 10% sessions than in the bottom 10% sessions.

Sequence logo analysis : Sequence logos are an efficient visualization tool for representing distributions of various observations over discrete time. For instance, they are used in biomedical research for visually representing sequences of genes. In our work, we used sequence logos to investigate the profile of dialogue modes in temporal space, i.e., as they unfold across throughout a dialogue session. The sequence logo regards each dialogue session as a discrete sequence of dialogue modes and then determines the dominant mode at each discrete moment in the sequence. The dialogue mode at the top of a stack of modes at each discrete moment of the dialogue is the most frequent mode at that moment. Furthermore, the height of each letter in a stack represents the amount of information contained. The bigger the letter/mode at a particular discrete time the more certain the

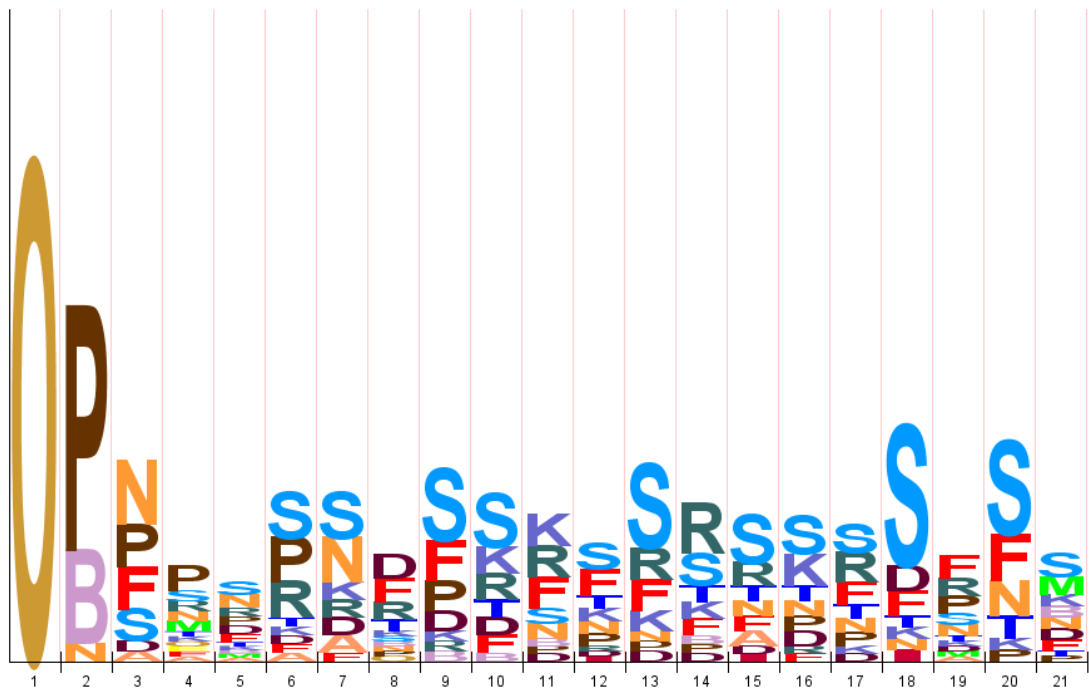


Fig. 2.5: Dialogue mode sequence logo for top 10% sessions up to average mode switch length of 21.

dominance of the corresponding mode is. For instance, at the discrete time 1 in the sequence logo shown in Figure 2.5 the dominant mode is *Opening*.

Figure 2.5 and Figure 2.6 show the sequence logo for the top 10% and bottom 10% sessions, respectively. We show the sequence logo diagram for the average mode switch length (21 for top 10% sessions and 11 for bottom 10% sessions). That is, we considered only those sessions having a number of mode switches greater or equal to the average mode switch length. The sessions with a larger number of mode switches were truncated to the average mode switch length.

From the sequence logos thus generated, we can infer the most certain sequence of dialogue modes in a typical human tutoring session which would be the sequence of the most certain dialogue modes at each discrete moment. The dominant sequence of modes is O, P, N, P, S, S, S, D, S, S, K, S, S, R, S, S, S, S, F,

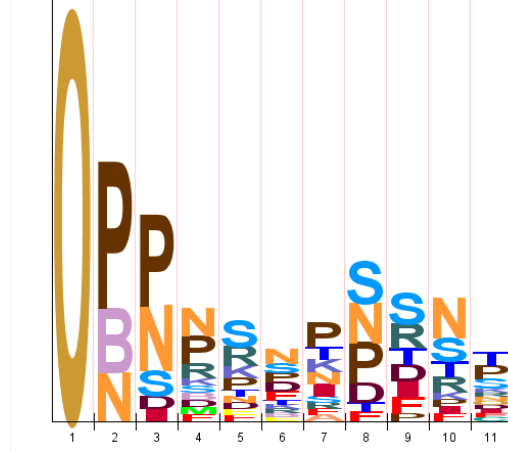


Fig. 2.6: Dialogue mode sequence logo for bottom 10% sessions up to average mode switch length of 11.

S and S for the top sessions as illustrated in Figure 2.5. On the other hand, the bottom sessions (Figure 2.6) are characterized by the following dominant sequence of modes/logos: O, P, P, N, S, N, P, S, S, N and T. In both figures, the symbols represent the dialogue modes as shown in Table 2.7.

The sequence logos reinforce the observations inferred from the profile comparison analysis. *Scaffolding* is the most frequent dominant mode in the dominant sequence for the top 10% sessions. Moreover, the dominant sequence for the top 10% contains *Modeling*, *RoadMap* and *Fading* modes that are absent in the dominant sequence for the bottom 10% sessions. Also, *Fading* and *RoadMap* appear as the second most dominant modes in two or three different time instants next to *Scaffolding*.

On the other hand, the dominant sequence for the bottom 10% sessions comparatively contains more *ProcessNegotiation* and *ProblemIdentification* modes than in the dominant sequence for the top 10% sessions. Another interesting observation is that the bottom 10% sessions are significantly shorter than top 10% sessions in terms of the average number of mode-switches (21 for top 10% sessions and 11 for bottom 10% sessions). All these observed patterns provide further evidence reported by previous studies that good tutors quickly identify gaps in

SN	Subsequence	p-value	Freq-bottom	Freq-top
1	(T-Expressive)-(T-Expressive)	0.0003	0.55	0.94
2	(T-Assertion)-(T-Expressive)	0.0003	0.61	0.97
3	(S-Expressive)-(T-Prompt)- (T-Expressive)	0.0005	0.16	0.66
4	(S-Assertion)-(T-Expressive)- (T-Expressive)	0.0010	0.29	0.77
5	(T-Assertion)-(S-Expressive)- (T-Expressive)	0.0043	0.27	0.73
6	(S-Expressive)-(T-Expressive)	0.0051	0.63	0.96
7	(S-Expressive)-(T-Expressive)- (T-Expressive)	0.0101	0.18	0.63
8	(S-Expressive)-(S-Expressive)	0.0103	0.41	0.83
9	(S-Assertion)-(T-Expressive)- (T-Prompt)	0.0271	0.27	0.71
10	(T-Prompt)-(T-Expressive)- (S-Expressive)	0.0638	0.16	0.58
11	(S-Assertion)-(T-Expressive)	0.0658	0.67	0.96
12	(T-Expressive)-(S-Expressive)- (T-Expressive)	0.1017	0.39	0.78

Table 2.8: Top 12 discriminant speaker differentiated act subsequences.

tutees’ knowledge and focus on encouraging the tutees, through the use of *Expressives* and *Scaffolding*, to solve the target problem by providing more targeted collaborative support.

Discriminant Sub-sequence analysis: Further, we investigated distinctive sub-sequences of dialogue acts and modes that are associated with effective and less effective sessions. In order to do this, we categorized all human annotated sessions having ES and EL scores ≤ 2 as ineffective, and all sessions rated with ES = 5 and EL ≥ 4 as good or effective sessions. Here, we used this categorization instead of top 10% vs. bottom 10% to include additional good and bad sessions that might have been excluded because of the restriction on the number of sessions imposed by the latter criteria. Using more good and bad sessions allows us to generate a large number of different sub-sequences for the analysis.

We then conducted sequence pattern mining using *Traminer* package in R.

The *Traminer* algorithm first finds the most frequent sub-sequences by counting their distinct occurrences and then applies a Chi-squared test (Bonferroni-adjusted) to identify sub-sequences that are statistically more (or less) frequent in each group. We used a p-value < 0.4 threshold instead of typical 0.05 or 0.1 value because this gave us a sufficient number of likely distinctive sub-sequences of modes which we used for deriving a tutorial Markov process which we describe later. However, for our discriminant sub-sequence analysis, we made the inferences/conclusions using only sub-sequences with p-value ≤ 0.1 threshold. Therefore, our conclusions would not change from the conclusions derived from such analysis done at the 0.1 significance threshold.

We used dialogue acts, act-subacts and mode-switches as observations. We also granularized the observations further by adding speaker information. Here, we report the most interesting sub-sequences discovered with this analysis. It should be noted that a sub-sequence is not necessarily a contiguous sequence of observation, however, the order of the observations is preserved. For example, *(Assertion)-(Expressive)* is a valid sub-sequence of dialogue acts formed from the *(Assertion)-(Request)-(Expressive)* contiguous sequence fragment. We generated sub-sequences up to length 7 from all annotated tutorial sessions.

The discriminant sub-sequences thus obtained (as shown in Table 2.8) further support the observations derived earlier based on the dialogue act profile comparison which indicated that good, i.e. effective, tutors use more *Expressive* and *Prompts*. That is, more prompting of students and more feedback through *Expressives* are signatures of effective sessions. We notice that all discriminant sub-sequences of acts contain *Expressive* acts initiated by either tutors or students. The good tutors often prompt students to acknowledge that they are following their tutoring or to elicit further answers or reasoning from the students. The discriminant sub-sequence analysis for act-subacts in Table 2.9 provide further

insights. Tutors' expressions of praise (*T-Expressive-Positive*) and farewell (*T-Expressive-Farewell*) and students expressing thanks (*S-Expressive-Thanks*) are highly predictive of effective sessions. The tutors often praise students to keep them engaged in the task and when students provide correct answers. The tutees expressing thanks (*S-Expressing-Thanks*) might suggest that the tutees are satisfied with the tutoring. Moreover, the tutor expressing farewell indicates that the tutoring session ends on a positive note. Sessions with proper closings might also suggest that both the student and the tutor are satisfied with the tutorial session.

The discriminant subsequent analysis for modes (Table 2.10) reveals interesting pedagogical patterns as well. Consistent with observations from the sequence logo analyses (Figure 2.6 and Figure 2.5) and the dialogue act and mode profile comparison, good tutorial sessions have *Scaffolding* and *Fading* as the dominant strategies i.e. the good tutors do more *Scaffolding* and *Fading* to encourage students to solve the problem by themselves, with minimal support. The sub-sequences S-S, F, S-F, F-F are very strong indicators of good sessions (p-value<0.05) while F-S, F-F-S, S-F-S also fairly strongly indicate sessions of top quality. Another interesting observation is that the Closing mode (p-value=0.0475) is also a very strong indicator of top sessions. Moreover, the Fading-Closing (p-value=0.002) sub-sequence is even more predictive than the Closing mode alone. We also observe that switching to *Scaffolding* or *Fading* modes after *ProblemIdentification* indicates effective tutoring as evidenced by the sub-sequences O-P-F (p-value=0.1764), P-F (p-value=0.0198) and P-S-S (p-value=0.0362).

Tutorial Markov Process: We used discriminant mode sub-sequences generated above to generate a Markov process for effective tutorial sessions. We first created the state transition matrix as follows.

We ignored sub-sequences of unit length as they don't indicate an observed transition. For sub-sequences spanning more than two states, we split it into

SN	Subsequence	p-value	Freq-bottom	Freq-top
1	(T-Expressive-Positive)	0.0001	0.21	0.72
2	(S-Expressive-Thanks)	0.0108	0.37	0.79
3	(T-Expressive-Farewell)	0.0171	0.27	0.70
4	(T-Prompt-Question)	0.1584	0.15	0.55
5	(S-Assertion-Calculation)- (T-Assertion-Calculation)	0.1950	0.17	0.56

Table 2.9: Top 5 discriminant speaker differentiated act-subact subsequences.

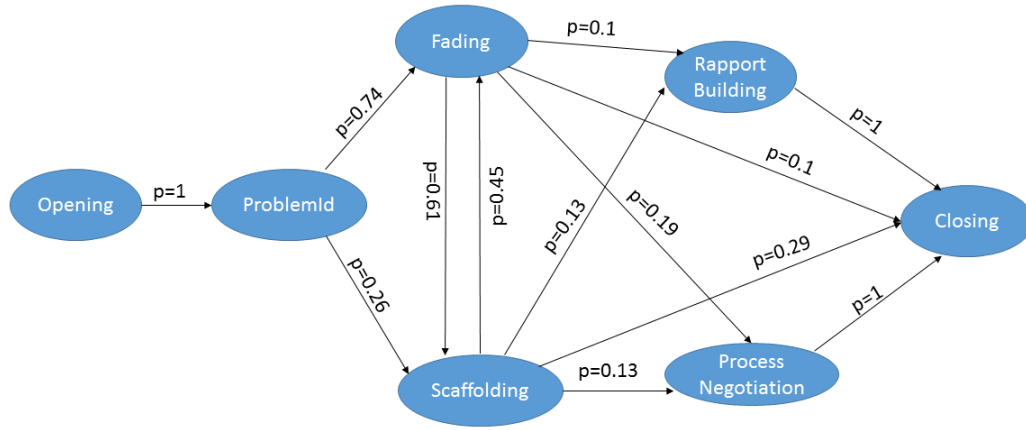


Fig. 2.7: Tutorial Markov Process for effective tutorial sessions.

multiple bi-gram sub-sequences, i.e., sub-sequences involving two consecutive states. For example, we obtained the bigram sub-sequences O-P and P-F from the O-P-F sub-sequence.

We also discarded self-transition paths since we consider modes as the states. It should be noted that modes are actually mode switches in our case since the subject matter experts (SMEs) annotated data only at utterances where mode switches occurred. For instance, they annotated an utterance with the dialogue mode label of *ProblemIdentification* where a switch occurred from, say, an *Opening*

SN	Subsequence	p-value	Freq-bottom	Freq-top
1	F-C	0.0002	0.137	0.638
2	S-S	0.0008	0.314	0.775
3	F	0.0009	0.451	0.871
4	S-F	0.0055	0.333	0.767
5	F-F	0.0132	0.176	0.612
6	P-F	0.0198	0.333	0.75
7	P-S-S	0.0362	0.176	0.598
8	C	0.0475	0.529	0.879
9	F-B	0.0511	0.078	0.4741
10	F-N	0.0643	0.333	0.732
11	F-S	0.0667	0.294	0.698
12	F-N-C	0.0676	0.117	0.517
13	F-F-S	0.0780	0.078	0.465
14	P-F-S	0.1039	0.117	0.508
15	F-S-F	0.13811	0.137	0.526
16	S-C	0.1467	0.235	0.629
17	O-P-F	0.1764	0.159	0.543
18	B	0.1788	0.353	0.733
19	F-S-C	0.1845	0.019	0.362
20	S-F-S	0.2003	0.216	0.603
21	S-B	0.2302&0.117	0.491	
22	F-S-N	0.2607	0.157	0.534
23	B-C	0.3723	0.157	0.526

Table 2.10: Discriminant mode subsequences. Symbols in subsequences represent dialog modes as described in Table 2.7.

to the *ProblemIdentification* mode. Therefore, we considered the transition path P-S but not S-S in the P-S-S sub-sequence.

We then computed the confidence score of an edge or path as the difference of 1 and the p-value of the sub-sequence the path belongs to. For example, for O-P-F (0.18), the confidence score of paths O-P and P-F is $1 - 0.18 = 0.82$. We weighted an edge as the cumulative sum of its confidence scores from all the sub-sequences where the path is present. For example, path P-F is present in sub-sequences P-F (0.02) and O-P-F (0.18) sub-sequences. So, the weight of the path P-F is: $0.98 + 0.82 = 1.8$. Finally, we normalized the weight of each path A-B by dividing it by the sum total of the weights of all possible transitions from A. For

example, the weight of the path P-F is normalized by dividing it by the sum total of weights of all the transitions from P state.

Once the state transition matrix was identified, we generated a tutorial Markov process as shown in Figure 2.7, where the states are dialogue modes, and transitions are generated using only the discriminant sub-sequences of modes. In the Figure 2.7, each path has been labeled with the corresponding transition probabilities.

This Markov representation reveals that any sequence of modes generated by the Markov process and which starts with an *Opening* and ends with a *Closing* state is likely to have a large number of cyclical *Scaffolding - Fading* patterns. This result partly supports theoretical expert tutoring models based on the *modeling-scaffolding-fading* paradigm (Rogoff & Lave, 1984). The high occurrences of these modes provide evidence that effective tutors do engage students more and provide help only when needed. Cade et al. (Cade et al., 2008) also found that *Scaffolding* was a highly occurring mode in expert tutoring. They found a relatively low occurrence of the *Fading* mode, which they suggested might be explained by time constraints, i.e., the tutoring session prevented the tutors from spending too much time in the *Fading* mode. It should also be noted that their data were related to both Math and Science subjects while our data was related to Math only.

The Markov process also resembles Graesser's (Graesser et al., 1995) 5-step dialogue framework, which captures the tutorial phases prevalent in one-on-one tutoring: *i) Tutor asks question, ii) Student answers question, iii) Tutor gives short feedback, iv) Tutor and student collaboratively improve the quality of the answer, v) Tutor assesses student's understanding.* One probable effective tutorial path from the Markov process, which might be comparable to Graesser's framework, is *Opening - ProblemId - Fading - Scaffolding - Fading - ProcessNegotiation - Closing.*

Indeed, the sub-path *ProblemId* - *Fading* - *Scaffolding* - *Fading* - *ProcessNegotiation* resembles Graesser’s 5-step framework.

The first 3 phases in Graesser’s framework don’t align with the initial modes of the suggested learning path. This might be because of the difference in the tutoring environment. Graesser assumed tutor-driven sessions, which start by a tutor first asking a question or presenting a problem to be solved followed by a student answer, etc. In our case, the sessions are initiated by students who are seeking help from the tutors on specific problems. The tutor works together with the student to understand the problem (*ProblemId*). Then, the tutor fades, allowing the student to work on the problem by herself (*Fading*). The tutor may switch between *Scaffolding* and *Fading* to provide help (*Scaffolding*), only when needed. In this sense, the last two elements in Graesser’s framework can be considered to be aligned with the *Scaffolding* - *Fading* pattern.

The additional benefit of this Markov process representation is that it suggests multiple possible paths or meta-strategies that can lead to learning gains.

2.6.5 Analysis using computer measure of learning gain

After using human-rated measures of learning gains, we used a measure generated by Cognitive Tutor called the number of assists per minute to measure learning gain. The number of assists measures the level of help a student needs while learning with the help of Cognitive Tutor. We obtained the level of help a student needed in the Cognitive Tutor session right before the human tutor session as well as the level of help needed in the Cognitive Tutor sessions right after the human tutor session. A drop in the level of help needed which we refer now onwards by the term *net assists per minute*, is considered as evidence of progress or learning gains. We considered both human-tagged and machine-tagged sessions and used *net assists per minute* to conduct the following tutorial analyses.

Profile comparison analysis: We compared the profiles of the top 25%

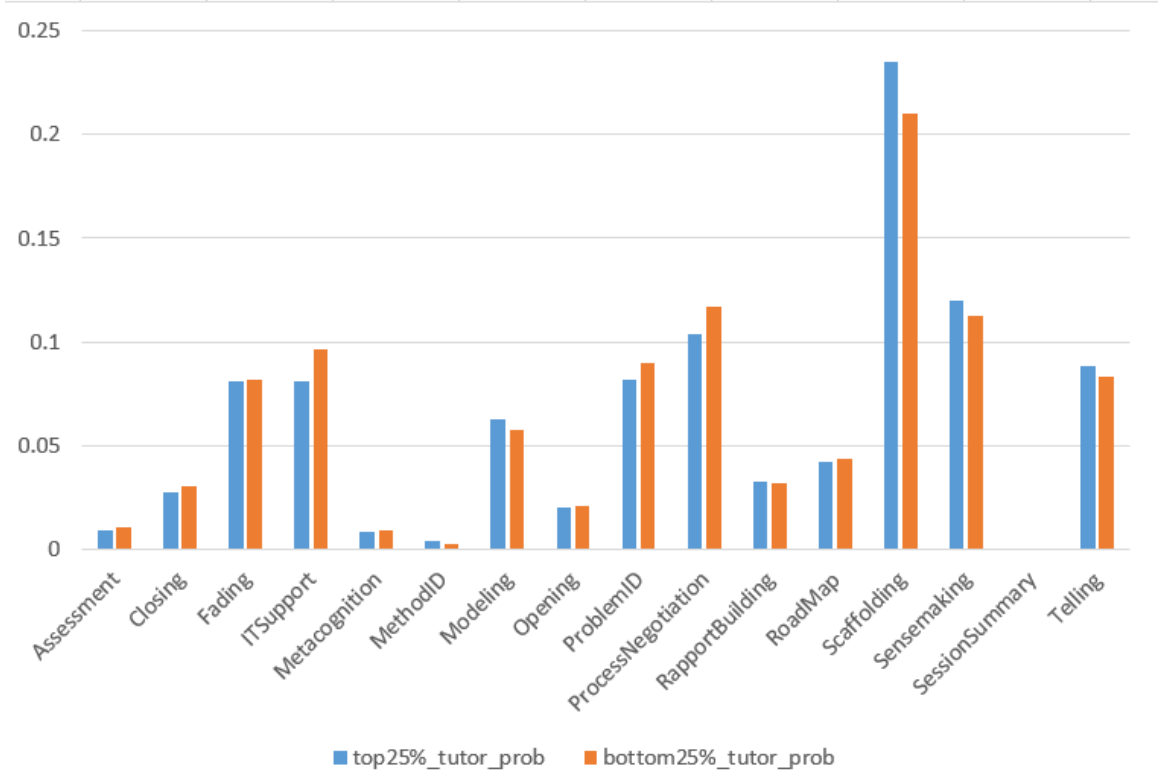


Fig. 2.8: Dialogue mode profiles of top versus bottom 25% sessions for tutors only.

versus the bottom 25% sessions in terms of learning gains. Figure 2.8 shows an example comparison of two dialogue mode profiles corresponding to top 25% versus bottom 25% of sessions when decreasingly ranked based on learning gains measured as the number of assists per minute. A closer analysis of the two profiles revealed that in the top sessions there are relatively more Modeling, Telling and Scaffolding modes triggered by tutors on average. In the bottom sessions, there are relatively more *ITSupport*, *ProblemId* and *ProcessNegotiation* modes initiated by tutors. Similarly, Figure 2.9 shows the dialogue mode profiles of the top 25% vs. bottom 25% where we account for both tutors and students. The profile shows that the top sessions are characterized by relatively more *Scaffolding* and less *ProcessNegotiation*.

Sequence logo analysis : We also conducted a sequence logo analysis for the top 25% and bottom 25% of the sessions based on *net assists per minute*. In this case, we show the sequence logo by using the tutorial sessions that have a number of

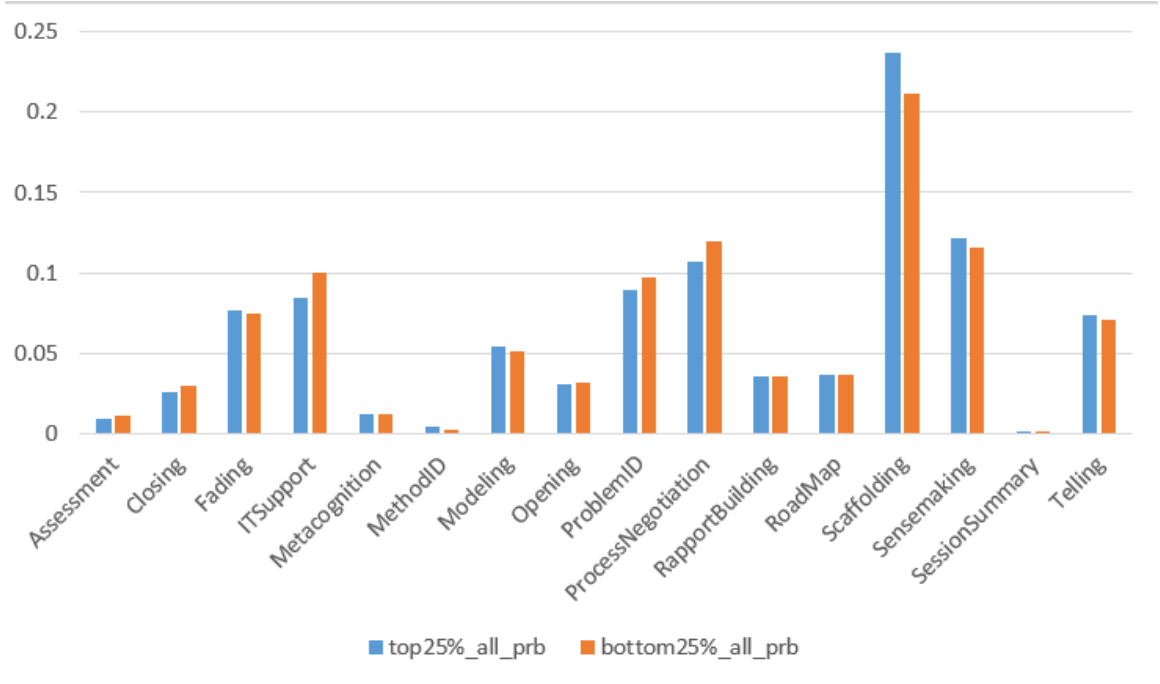


Fig. 2.9: Dialogue mode profiles of top versus bottom 25% sessions including both tutor and student initiated modes.

mode switches exactly equal to average mode switch length (20 for top 25% sessions and 19 for bottom 25% sessions). The dominant sequence of modes/logos is O, P, N, S, S, S, R, S, T, S, T, S, S, S, T, N, S, B, N, C for the top sessions as illustrated in Figure 2.10. On the other hand, the bottom sessions (Figure 2.11) are characterized by the following dominant sequence of modes/logos: O, P, N, N, N, N, K, S, S, T, S, K, S, T, K, N, K, N, C. In both figures, the symbols represent: O - Opening, P - Problem Identification, N - ProcessNegotiation, K - Sensemaking, S - Scaffolding, T - Telling, F - Fading, B - RapportBuilding and C - Closing. Interestingly, *Scaffolding* is the most frequent in the top sessions while *ProcessNegotiation* is the most frequent dominant dialogue mode in the bottom sessions. This observation is consistent with sequence logo analysis using human judged session quality scores i.e., an average of EL and ES.

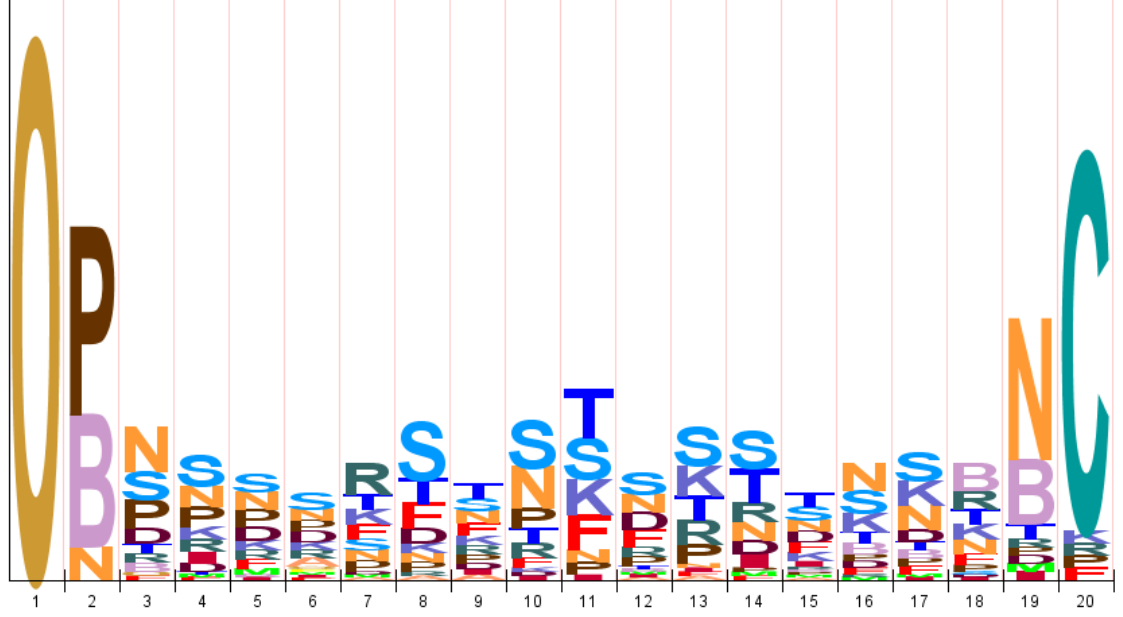


Fig. 2.10: Dialogue mode sequence logo for top 25% sessions of average length 20.

2.7 Conclusion

In this chapter, we presented our supervised approach to mapping the dialogue-based tutorial sessions onto dialogue acts, sub-acts and modes. We used both expert human judgments and an objective learning gains measure derived from students' interaction with a computer tutor. We presented our various analytic approaches such as profile comparison, sequence logo, discriminant sub-sequence mining, and Markov analysis to identify effective tutorial strategies.

We found that the effective tutorial sessions are characterized by more *Scaffolding* and *Fading* modes on average when compared to ineffective sessions. Furthermore, the most effective sessions almost always end properly, i.e., with a *Closing* mode. On the other hand, the bottom ineffective sessions have, on average, more *ProcessNegotiation* and *ProblemIdentification*. At dialogue act level, tutors in top sessions use more expressives and prompt students more, on average, than those in the bottom sessions.

We also found that any possible sequence of dialogue modes derived from

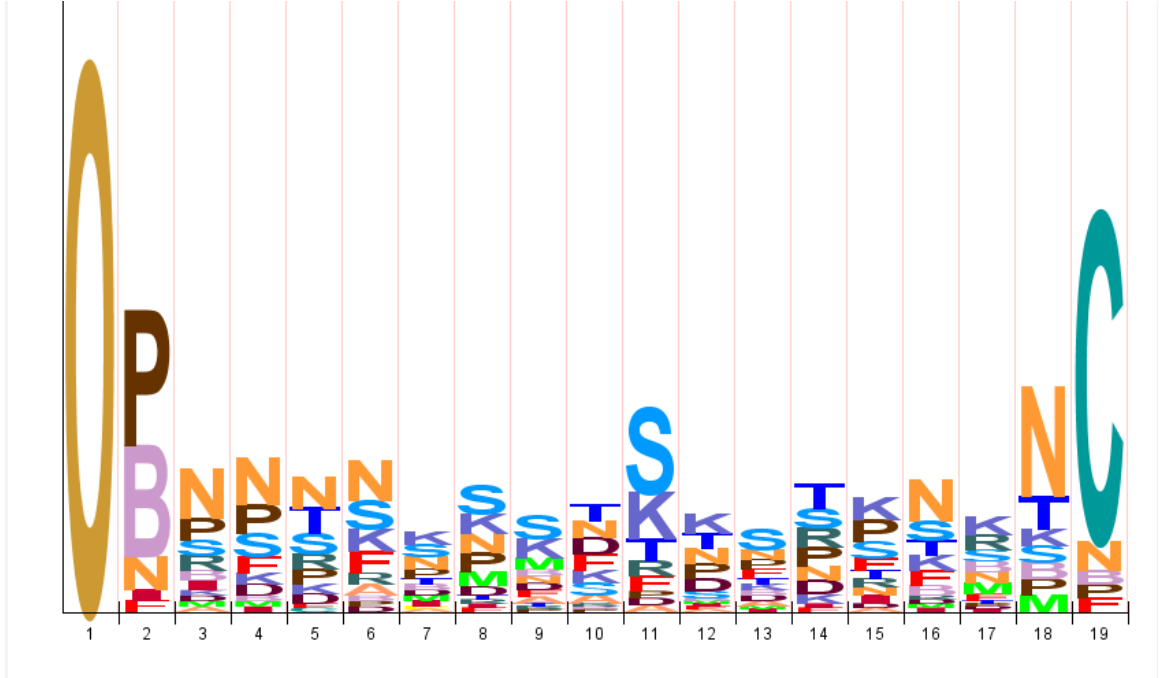


Fig. 2.11: Dialogue mode sequence logo for bottom 25% sessions of average length 19.

inferred Markov process which characterizes an effective tutorial session is most likely to have more *Scaffolding* and *Fading* modes. Further, the inferred Markov process suggests a new model for student-initiated tutorial sessions as opposed to tutor-driven sessions, which were modeled in the past.

Our future work is to expand our understanding of the effective strategies for effective tutorial sessions while accounting for other factors such as students' prior knowledge.

Chapter 3

Assessing Semantic Textual Similarity

One of the research questions we address in this work is to improve the quality of student answer assessment. Automatically assessing student answer requires an understanding of the meaning of the text. The task is one of the hardest problems in the Natural Language Processing field since it requires a large amount of world knowledge, domain knowledge, contextual knowledge, and linguistic knowledge as well as reasoning mechanisms to infer over these vast collections of knowledge to correctly assess the semantics (or meaning) of the text. For example, the sentence “*I saw her duck*” has many possible meanings. One interpretation is that “*I saw a duck that belonged to her.*” Another possible meaning is: “*I saw her squatting down to avoid something.*” The sentence can also mean “*I cut her duck with a saw.*” To correctly interpret its meaning, more information such as context, linguistic, etc. is required.

As such, the pragmatic approach to understanding the meaning of the text is semantic similarity. The task of semantic similarity is to measure how similar two texts are with a similarity score. In this work, we present our semantic similarity approach to improving the quality of student answer assessment. The student answers are usually short, spanning one or two sentences. We apply a similarity score scale that ranges from 0 (no relation) to 5 (semantic equivalence). Examples of pairs of sentences with different human-rated similarity scores and their interpretations based on this [0,5] scale is presented in Table 3.1.

Sentence-to-sentence similarity is one of the important semantic similarity tasks and has received an increasing amount of attention in recent years. Semantic textual similarity has a wide range of applications, such as text summarization (Steinberger & Jezek, 2004), plagiarism detection (Potthast et al., 2012), automated

Table 3.1: Interpretation of similarity score (SS) with corresponding example sentence pairs (Agirre et al., 2015).

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

answer assessment (Graesser, Penumatsa, Ventura, Cai, & Hu, 2007; Mohler & Mihalcea, 2009), question answering (Tapeh & Rahgozar, 2008), machine translation (Papineni, Roukos, Ward, & Zhu, 2002), dialog and conversational systems (Rus, Niraula, & Banjade, 2015). Therefore, many Semantic Evaluations (SemEval) tasks for measuring semantic textual similarity (STS) between two sentences have been carried out (Agirre, Diab, Cer, & Gonzalez-Agirre, 2012; Agirre, Cer, Diab, Gonzalez-Agirre, & Guo, 2013; Agirre et al., 2014, 2015; Agirre, Banea, et al., 2016; Cer et al., 2017). We have participated in the SemEval shared tasks for measuring semantic textual similarity in the last three years, 2015, 2016 and 2017.

We developed STS system called NeRoSim (Banjade, Niraula, et al., 2015) and DTSim (Banjade, Maharjan, Gautam, & Rus, 2016) for measuring sentence

similarity in the SemEval 2015 and 2016 competitions respectively. The NeRoSim system (Banjade, Niraula, et al., 2015) was ranked 4th best team in the SemEval 2015 competition but with no significant difference with the results of top performing systems (Agirre et al., 2015). The DTSim system (Banjade, Maharjan, Gautam, & Rus, 2016) was one of the top performing systems that correlated up to 0.83 for some datasets in the SemEval 2016 test dataset (Agirre, Banea, et al., 2016).

Recently, we developed a system called DT_TEAM (Maharjan, Banjade, Gautam, et al., 2017) which was placed second in the SemEval 2017 challenge for English - Track 5 (Cer et al., 2017). The correlation between our system’s output and human judgments was 0.8536 and almost as good as the best performing system which was at 0.8547 correlation. Our system also performed well (correlation = 0.792) when evaluated with a separate STS benchmark dataset (Cer et al., 2017; Maharjan, Banjade, Gautam, et al., 2017).

The outline of this chapter is as follows. Next, we discuss on related work (Section 3.1) followed by *Preprocessing* (Section 3.2) and *Approach* (Section 3.3). Then, we discuss our various DT_TEAM models and their results in the *Experiment and Results* (Section 3.4).

3.1 Related Work

Many research works have been conducted to measure the similarity between two sentences. Most methods for computing sentence similarity exploit word-to-word similarity measures or word/phrase vector representations. We discuss below word-to-word similarity and sentence-to-sentence similarity techniques.

3.1.1 Word-to-word Similarity

There are many different methods to measure the word-to-word similarity. Such methods can be broadly grouped under two categories based on the resources used. The first group of methods consists of knowledge-based methods which exploit structured semantic networks or ontologies such as WordNet (Miller, 1995).

Wordnet groups content words (nouns, verbs, adjectives, and verbs) with the same meaning that express a distinct concept into sets called synsets. The synsets are linked to other synsets by means of a limited number of conceptual relations such as hypernymy (super-subordinate relation, i.e., car is a hypernym of Honda), hyponymy (ISA relation, i.e., Honda is a hyponym of car), meronymy (a part-whole relationship, i.e., wheel is a meronymy of car), antonymy (semantically opposite relation, i.e., lack and abundance are antonyms) and synonymy (semantically equivalent relation, i.e., lack and scarcity are synonyms). Many methods based on Wordnet have been developed to compute similarity between two words. Some methods exploit distance-based measures on the network’s path (Leacock & Chodorow, 1998; Ho Lee, Ho Kim, & Joon Lee, 1993), some methods further account for the Information Content of the lowest common subsumer in the hierarchy (Jiang & Conrath, 1997; Lin et al., 1998) and some methods make use of WordNet gloss (a brief definitions of a concept) overlap (Pedersen, Patwardhan, & Michelizzi, 2004) for computing the word-to-word similarity.

The other group of methods consists of corpus-based methods that learn vector representation of words by exploiting the word associations or word co-occurrences from a large collection of texts such as Wikipedia or Google news in an unsupervised manner. These methods are also known as distributional or distributed methods. Once the vector representations are learned, the similarity between two words can be obtained by computing the cosine of their vectors. Some methods are purely algebraic such as Latent Semantic Analysis (Landauer, Foltz, & Laham, 1998) and Hyperspace Analogue to Language (Burgess & Lund, 1995) which represent words with low dimensional vectors by utilizing low-rank approximations. Stănescu, Banjade, and Rus (2014) developed LSA representations of words from the whole Wikipedia article. Bengio, Ducharme, Vincent, and Jauvin (2003) and Collobert et al. (2011) obtained vector representations of words by using deep

neural networks. Further, Mikolov, Chen, Corrado, and Dean (2013) developed a single hidden layer neural network for learning word representations based on two models: continuous bag of words (CBOW) and skip gram models. Pennington, Socher, and Manning (2014) learned word representations by training on aggregated global word-word co-occurrence statistics from a corpus. Further, a lexical paraphrase database known as PPDB (Ganitkevitch, Van Durme, & Callison-Burch, 2013; Pavlick, Rastogi, Ganitkevitch, Van Durme, & Callison-Burch, 2015) has been developed containing more than 8 million lexical paraphrase pairs. Each paraphrase pair is scored with a similarity score computed as the cosine of their context vectors. The context vector is a set of contextual features that a word/phrase occurs in such as n-grams seen to the left and right of the phrase.

Moreover, different ensemble word similarity approaches have been developed to measure word-to-word similarity (Jiang & Conrath, 1997; Li, Bandar, & McLean, 2003). Banjade, Maharjan, Niraula, Rus, and Gautam (2015) combined various knowledge-based and corpus-based methods using a regression model to measure word similarity. Similarly, (Niraula, Gautam, Banjade, Maharjan, & Rus, 2015) combined different vector representations for measuring word similarity.

3.1.2 Sentence-to-Sentence Similarity

The basic approach to measure sentence similarity is a simple word overlap measure which computes the similarity between the two sentences as the proportion of words that appear in the two sentences normalized by the sentence length. Another variant of word overlap measure is the Jaccard coefficient which computes the sentence similarity score as ratio of the size of the intersection of the words in the two sentences to the union of the words in the two sentences. The approach is further improved by considering lemmatization, stop-word removal, and part-of-speech categories. However, this approach ignores the relationship between the two words if they are not lexically identical. The words might be related to each



Fig. 3.1: General Pipeline of DT_TEAM System.

other with various semantic relations such as synonymy, hypernymy, antonymy etc. Moreover, words co-occurring frequently should be more similar than words that co-occur with less frequency. Therefore, many approaches that utilize word-to-word similarity have been developed that improved the similarity of the texts. Fernando and Stevenson (2008) used various WordNet-based word similarity measures to compute the sentence similarity while Mihalcea, Corley, Strapparava, et al. (2006) used both corpus-based and knowledge-based word-to-word similarity measures for computing sentence similarity.

The most popular approach is the alignment approach where words/phrases are aligned across the two sentences, and then, the sentence similarity score is computed by computing the similarity scores between the aligned pairs. The alignment approach is intuitive as high similarity score indicates the alignment of many highly semantically similar words. BLEU which is a commonly used measure in machine translation (Papineni et al., 2002) is based on word/phrase alignment. Rus and Lintean (2012) applied greedy and optimal word alignment methods with WordNet and LSA based word-to-word similarity measures. Another method aligned the sentences at the chunk level and computed sentence similarity by utilizing chunk similarity scores (Ștefănescu, Banjade, & Rus, 2014). Banjade, Maharjan, Gautam, and Rus (2016) computed sentence similarity by aligning the chunks across the two sentences by various semantic relations and semantic scores. The alignment-based methods have proved to be better at measuring semantic textual similarity (Agirre et al., 2015; Agirre, Banea, et al., 2016; Banjade, Maharjan, Gautam, & Rus, 2016; Sultan, Bethard, & Sumner, 2015).

Another approach is to learn the vector representation for sentences directly and compute the cosine score to measure the sentence similarity. However, learning meaningful sentence representations directly is difficult due to the sparseness problem. Therefore, the common approach is to learn sentence representations by composing the word vectors. A simple composition method is to simply add the vectors of the words in a sentence and then compute cosine between the resultant vectors to measure sentence similarity. A variant of algebraic resultant vector representation adds the words weighted by their parts-of-speech categories (Maharjan, Banjade, Gautam, et al., 2017).

Recently, many deep neural network-based composition methods have been developed which learn the sentence representations from the word/phrase representations using recursive networks (Socher et al., 2013), recurrent neural networks (Hochreiter & Schmidhuber, 1997), convolutional networks (Kalchbrenner, Grefenstette, & Blunsom, 2014) and recursive-convolutional methods (Zhao, Lu, & Poupart, 2015). Microsoft recently developed Sent2Vec tool that used both Deep Structured Semantic Model (DSSM) network (Huang et al., 2013) and DSSM with convolutional-pooling (CDSSM) network (Shen, He, Gao, Deng, & Mesnil, 2014; Gao, Deng, Gamon, He, & Pantel, 2014) to generate the continuous vector representations for sentences. Further, Kiros et al. (2015) used RNN GRU encoder - RNN decoder model for learning generic sentence representations.

Moreover, there are machine learning methods that combine various features such as n-grams, word overlap, sentence similarity scores computed using different methods, etc. for measuring sentence similarity (Banjade, Niraula, et al., 2015; Maharjan, Banjade, Gautam, et al., 2017). There are also ensemble methods that incorporate various feature engineered systems including deep learning models (Tian, Zhou, Lan, & Wu, 2017). These ensemble systems and featured engineered systems have proved to be top performing systems in recent STS

competitions (Agirre, Banea, et al., 2016; Cer et al., 2017). The ECNU system (Tian et al., 2017) which was ranked first overall in the STS 2017 competition (Cer et al., 2017) was an ensemble system while our DT_TEAM was a feature engineered system that ranked second in STS 2017 challenge for the English track.

We describe our DT_TEAM system in this chapter. The general pipeline of the system is shown in Figure 3.1. We describe the preprocessing component in the next section.

3.2 Preprocessing

The preprocessing steps were same as our DTSim system (Banjade, Maharjan, Gautam, & Rus, 2016). We first removed hyphens in the hyphenated words except when they were composite verbs (e.g., video-gamed) or started with co-, pre-, meta-, multi-, re-, pro-, al-, anti-, ex-, and non- prefixes. Then, we applied tokenization, lemmatization, POS-tagging, name-entity recognition using Stanford CoreNLP Toolkit (Manning et al., 2014). We also marked the word tokens if they are stop-words or not. Further, we created normalization look-up resources to normalize words to a single representation. For example, we normalized all occurrences of *pc*, *pct*, *%* to *pc*. We also created chunks using our own Conditional Random Fields (CRF) based chunking tool (Maharjan, Banjade, Niraula, & Rus, 2016). The chunking tool is described in detail in the next Chapter 5.

3.3 Approach

Our DT_TEAM (Maharjan, Banjade, Gautam, et al., 2017) is a feature engineered system that includes deep learning signals using sentence embeddings from DSSM, CDSSM and skip-thought models. Our system includes three different models: Support Vector Regression (SVR), Linear Regression (LR) and Gradient Boosting Regressor (GBR) models. The engineered features include similarity scores calculated using word and chunk alignments, unigram overlap, a fraction of unaligned words, difference in word counts by type (all, adj, adverbs, nouns, verbs),

and min to max ratios of words by type. We next discuss how different features were generated and used in developing different models in our system.

3.3.1 Feature Generation

We generated various features including similarity scores generated using different methods. We describe next the word-to-word and sentence-to-sentence similarity methods used to generate the features.

Word Similarity Methods

We used word2vec (Mikolov et al., 2013)¹ vectorial word representation, PPDB database (Pavlick et al., 2015)², and WordNet (Miller, 1995) to compute similarity between words. As described in (Banjade, Maharjan, Gautam, & Rus, 2016), we used the following Equation 3.1 to compute word similarity score.

$$sim(w_1, w_2, m) = \begin{cases} 1, & \text{if } w_1 \text{ and } w_2 \text{ are synonyms} \\ 0, & \text{if } w_1 \text{ and } w_2 \text{ are antonyms} \\ ppdb(w_1, w_2), & \text{if } m = ppdb \\ cosine(x_1, x_2), & \text{otherwise} \end{cases} \quad (3.1)$$

where $m \in \{ppdb, word2vec\}$ word representation model set, x_1 and x_2 are vector representations of words w_1 and w_2 respectively. We check if w_1 and w_2 have synonymy or antonymy relations or not using WordNet.

Sentence Similarity Methods

Word Alignment Method: We lemmatized all content words and aligned them optimally using the Hungarian algorithm (Kuhn, 1955) implemented in the SEMILAR Toolkit (Rus, Lintean, Banjade, Niraula, & Stefanescu, 2013). The process is the 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

¹<http://code.google.com/p/word2vec/>

²<http://www.cis.upenn.edu/~ccb/ppdb/>

Table 3.2: Example of chunk alignment between two short text with semantic labels and scores.

Example text pairs (plain)	
S_1 :	Bangladesh building disaster death toll passes 500
S_2 :	Bangladesh building collapse: death toll climbs to 580
Example text pairs (chunked)	
S_1 :	[Bangladesh building disaster][death toll][passes][500]
S_2 :	[Bangladesh building collapse][:][death toll][climbs][to 580]
Alignment Output	
1 2 3 \Leftrightarrow 1 2 3 //	EQUI // 5.0 // Bangladesh building disaster \Leftrightarrow Bangladesh building collapse
4 5 \Leftrightarrow 5 6 //	EQUI // 5.0 // death toll \Leftrightarrow death toll
7 \Leftrightarrow 8 9 //	SIMI // 3.0 // 500 \Leftrightarrow to 580
6 \Leftrightarrow 0 //	NOALI // 0 // passes \Leftrightarrow -not aligned-
0 \Leftrightarrow 4 //	NOALI // 0 // -not aligned- \Leftrightarrow :
0 \Leftrightarrow 7 //	NOALI // 0 // -not aligned- \Leftrightarrow climbs

word pairs computed as described in § 3.3.1. In order to avoid noisy alignments, we reset the similarity score below 0.5 (empirically set threshold) to 0. The similarity score was computed as the sum of the scores for all optimally aligned word-pairs (OAs) divided by the total length of the given sentence pair (S_1, S_2) as shown in Equation 3.2.

$$sim(S_1, S_2) = 2 * \frac{\sum_{(w_1, w_2) \in OA} sim(w_1, w_2)}{|S_1| + |S_2|} \quad (3.2)$$

Interpretable Similarity Method: We aligned chunks across sentence-pairs and labeled the aligned chunks with semantic score and semantic relations (Banjade, Niraula, et al., 2015; Maharjan et al., 2016). A chunk is a syntactically meaningful unit which typically consists of a single content word surrounded by a group of function words (Abney, 1991). Agirre et al. (2015) introduced a set of semantic relation types: EQUI (chunks are semantically equivalent), OPPO (chunks are opposite in meaning), SPE1/SPE2 (the chunk in the first/second sentence is more specific than the chunk in the second/first sentence), SIMI (chunks are similar but not EQUI, OPPO or SPE), REL (chunks are related but not EQUI, OPPO, SPE or SIMI), ALIC (a chunk is not aligned to any other chunk due to 1:1 alignment

restriction) and NOALIC (the chunk is unrelated and has no alignment). Table 3.2 shows an example of such alignment.

We used the Equation 3.3 as described by Banjade, Maharjan, Gautam, and Rus (2016) to compute the sentence similarity based on the set of semantically aligned chunked pairs (SA). $sim(c_1, c_2)$ is the similarity score between the chunks c_1 and c_2 .

$$sim(S_1, S_2) = \frac{\sum_{(c_1, c_2) \in SA} sim(c_1, c_2)}{5 * (\text{Total \# alignments including NOALI})} \quad (3.3)$$

Gaussian Mixture Model Method: The Gaussian Mixture Model (GMM) is widely used for soft clustering in an unsupervised manner. In our GMM approach, we assumed that a sentence pair might belong to any of the semantic clusters, i.e., semantic class levels [0, 5]. We represented the sentence pair as a feature vector consisting of feature sets {7, 8, 9, 10, 14} from Section 3.3.2 and modeled the semantic class levels as multivariate Gaussian densities of feature vectors. We first initialized the GMM parameters in a supervised manner since we have semantic class labels for the training data. We then used Expectation Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) for estimating the GMM parameters. We used the trained GMM to compute membership weights to each of these semantic levels for a given sentence pair. Finally, the GMM score (similarity score) is computed using Equation 3.4 and Equation 3.5. x is a feature vector representing a sentence pair. i represents a semantic class label. $N(x|\mu_i, \sum_i)$ is a multivariate Gaussian density function for semantic class label i .

$$mem_wt_i = w_i N(x|\mu_i, \sum_i), i \in [0, 5] \quad (3.4)$$

$$gmm_score = \sum_{i=0}^5 mem_wt_i * i \quad (3.5)$$

Compositional Sentence Vector Method: We used both Deep Structured Semantic Model (Huang et al., 2013) and DSSM with convolutional-pooling (Shen et al., 2014; Gao et al., 2014) in the Sent2vec tool³ to generate the continuous vector representations for given texts. We then computed the similarity score as the cosine similarity of their representations.

Tuned Sentence Representation Based Method: We first obtained the continuous vector representations V_A and V_B for sentence pair A and B using the Sent2Vec DSSM or CDSSM models or skip-thought model⁴ (Zhu et al., 2015; Kiros et al., 2015). Inspired by Tai, Socher, and Manning (2015), we then represented the sentence pairs by the features formed by concatenating element-wise dot product $V_A.V_B$ and absolute difference $|V_A - V_B|$. We used these features in our logistic regression model which produces the output \hat{p}_θ . Then, we predicted the similarity between the texts in the target pair as $\hat{y} = r^T \hat{p}_\theta$, where $r^T = \{1, 2, 3, 4, 5\}$ is the ordinal scale of similarity. To enforce that \hat{y} is close to the gold rating y , we encoded y as a sparse target distribution p such that $y = r^T p$ as:

$$p_i = \begin{cases} y - \lfloor y \rfloor, i = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, i = \lfloor y \rfloor \\ 0, \text{ otherwise} \end{cases} \quad (3.6)$$

where $1 \leq i \leq 5$ and, $\lfloor y \rfloor$ is *floor* operation. For instance, given $y = 3.2$, its sparse vector representation p derived from the Equation 3.6 is $[0 \ 0 \ 0.8 \ 0.2 \ 0]$. For building logistic model, we used training data set from our previous DTSim system (Banjade, Maharjan, Gautam, & Rus, 2016) and used image test data from STS-2014 and STS-2015 as validation data set.

Similarity Vector Method: We generated a vocabulary V of unique words from

³<https://www.microsoft.com/en-us/download/details.aspx?id=52365>

⁴<https://github.com/ryankiros/skip-thoughts>

the given sentence pair (A, B) . Then, we generated sentence vectors as in the followings: $V_A = (w_{1a}, w_{2a}, ..w_{na})$ and $V_B = (w_{1b}, w_{2b}, ...w_{nb})$, where $n = |V|$ and $w_{ia} = 1$, if $word_i$ at position i in V has a synonym in sentence A . Otherwise, w_{ia} is the maximum similarity between $word_i$ and any of the words in A , computed as: $w_{ia} = \max_{j=1}^{j=|A|} sim(w_j, word_i)$. The $sim(w_j, word_i)$ is the cosine similarity score computed using the word2vec model. Similarly, we compute V_B from sentence B .

Weighted Resultant Vector Method: We combined word2vec word representations to obtain sentence level representations through vector algebra. We weighted the word vectors corresponding to content words. We generated resultant vector for text A as $R_A = \sum_{i=1}^{i=|A|} \theta_i * word_i$, where the weight θ_i for $word_i$ was chosen as $word_i \in \{\text{noun} = 1.0, \text{verb} = 1.0, \text{adj} = 0.2, \text{adv} = 0.4, \text{others (e.g. number, which has CD POS tag)} = 1.0\}$.

Similarly, we computed resultant vector R_B for text B . We varied the weights from 0 to 1 for each class of words and obtained the best result with the above set of weights on the training data. We then calculated a similarity score as the cosine of R_A and R_B . Finally, we penalized the similarity score by the unalignment score which we describe next.

Penalization: We applied the following two penalization strategies to adjust the sentence-to-sentence similarity score.

Penalizing by Crossing Score: Crossing measures the spread of the distance between the aligned words in a given sentence pair. In most cases, sentence pairs with higher degree of similarity have aligned words in same position or its neighborhood. We define crossing score crs by Equation 3.7

$$crs = \frac{\sum_{w_i \in A, w_j \in B, aligned(w_i, w_j)} |i - j|}{\max(|A|, |B|) * (\#alignments)} \quad (3.7)$$

where $aligned(w_i, w_j)$ refers to word w_i at index i in A and w_j at index j in

B are aligned. Then, the similarity score was reset to 0.3 if $crs > 0.7$. The threshold 0.7 was empirically set based on evaluations using the training data.

Penalizing by Unalignment Score: We define unalignment score similar to word alignment score but this time the score is calculated using the unaligned words in both A and B as:

$$unalign_score = \frac{|A| + |B| - 2 * (\#alignments)}{|A| + |B|} \quad (3.8)$$

Then, the similarity score was penalized by using the Equation 3.9. The weight 0.4 in the equation was empirically chosen.

$$score^* = (1 - 0.4 * unalign_score) * score \quad (3.9)$$

3.3.2 Feature Selection

We generated and experimented with many features. We describe here only those features used directly or indirectly by our three runs submitted in the SemEval 2017 STS task for the English track. We describe the three runs in Section 3.4. We used word2vec representation and WordNet antonym and synonym relations for word similarity unless anything else is mentioned specifically.

1. $\{w2v_wa, ppdb_wa, ppdb_wa_pen_ua\}$: similarity scores generated using word alignment based methods (pen_ua for scores penalized by unalignment score).
2. $\{gmm\}$: sentence similarity score computed by using Gaussian Mixture Model method.
3. $\{dssm, cdssm\}$: similarity scores computed by generating sentence vector representations using both DSSM and CDSSM models.
4. $\{dssm_lr, skipthought_lr\}$: similarity scores using tuned sentence

representation based method. We used sentence representations from DSSM and skip-thought models.

5. $\{sim_vec\}$: score using similarity vector method.
6. $\{res_vec\}$: score using the weighted resultant vector method .
7. $\{interpretable\}$: score calculated using interpretable similarity method.
8. $\{noun_wa, verb_wa, adj_wa, adv_wa\}$: Noun-Noun, Adjective-Adjective, Adverb-Adverb, and Verb-Verb alignment scores using word2vec for word similarity.
9. $\{noun_verb_mult\}$: multiplication of Noun-Noun similarity scores and Verb-Verb similarity scores.
10. $\{abs_diff_t\}$: absolute difference as $\frac{|C_{ta}-C_{tb}|}{C_{ta}+C_{tb}}$ where C_{ta} and C_{tb} are the counts of tokens of type $t \in \{\text{all tokens, adjectives, adverbs, nouns, and verbs}\}$ in sentence A and B respectively.
11. $\{overlap_pen\}$: unigram overlap between text A and B with synonym check given by: $score = \frac{2*overlap_count}{|A|+|B|}$. Then penalized by both crossing and unalignment score.
12. $\{noali\}$: number of NOALI semantic chunk relations in aligning chunks between texts relative to the total number of alignments.
13. $\{align, unalign\}$: fraction of aligned/non-aligned words in the sentence pair.
14. $\{mmr_t\}$: min to max ratio as $\frac{C_{t1}}{C_{t2}}$ where C_{t1} and C_{t2} are the counts of type $t \in \{\text{all, adjectives, adverbs, nouns, and verbs}\}$ for shorter text 1 and longer text 2, respectively.

Table 3.3: Summary of training data.

Data set	Count	Release time
Deft-news	299	STS2014-Test
Images	749	STS2014-Test
Images	750	STS2015-Test
Headlines	742	STS2015-Test
Answer-forums	375	STS2015-Test
Answer-students	750	STS2015-Test
Belief	375	STS2015-Test
Headlines	244	STS2016-Test
Plagiarism	230	STS2016-Test
Total	4514	

3.4 Experiment and Results

3.4.1 Datasets

Training Data

We used a subset of the data released in previous STS competitions from the year 2014 to 2015 (see Table 3.3) for developing various models. The training dataset consists of expert annotated sentence pair instances from diverse data sources such as Images, Answer-forums, Belief, Headlines, Plagiarism, and Deft-news.

Test Data

The evaluation consisted of 250 sentence pairs from the Stanford Natural Language Inference data (Bowman, Angeli, Potts, & Manning, 2015).

STS Benchmark Data

The STS benchmark data (Cer et al., 2017) was released by the STS 2017 task organizers to provide a standard benchmark to compare among STS systems in future years. The data consisted of 8,628 sentence pairs carefully selected from the English datasets, namely, news headlines, image captions, and forum, used in previous STS tasks between 2012 and 2017. The benchmark data is divided into three subsets: Train, Dev and Test datasets. The distribution of the benchmark data is shown in Table 3.4.

Table 3.4: Distribution of STS benchmark data according to different genres and data partitions.

Genre	Train	Dev	Test	Total
news	3299	500	500	4299
caption	2000	625	525	3250
forum	450	375	254	1079
total	5749	1500	1379	8628

Table 3.5: Detailed breakdown of STS benchmark data by original names and task years of the datasets.

Genre	File	Year	Train	Dev	Test
news	MSRpar	12	1000	250	250
news	headlines	13/6	1999	250	250
news	deft-news	14	300	0	0
captions	MSRvid	12	1000	250	250
captions	images	14/5	1000	250	250
captions	track5.en-en	17	0	125	125
forum	deft-forum	14	450	0	0
forum	ans-forums	15	0	375	0
forum	ans-ans	16	0	0	254

The benchmark Train set can be used for training the model while the Dev set can be used for tuning the parameters of the model. The Test set is provided for evaluating the performance of the model. The detailed breakdown of the STS benchmark data by original names and task years of the datasets is provided in Table 3.5.

3.4.2 Models and Runs

Using the combination of features described in Section 3.3.2, we built three different models corresponding to the three runs (R1-3) submitted.

R1. Linear SVM Regression model (SVR; $\epsilon = 0.1$, $C = 1.0$) with a set of 7 features: *overlap_pen*, *ppdb_wa_pen_ua*, *dssm*, *dssm_lr*, *noali*, *abs_diff_all_tkns*, *mmr_all_tkns*.

R2. Linear regression model (LR; default weka settings) with a set of 8 features: *dssm*, *cdssm*, *gmm*, *res_vec*, *skiphought_lr*, *sim_vec*, *aligned*, *noun_wa*.

Table 3.6: Results of our submitted runs on test data (1st is the best result among the participants).

R1	R2	R3	Baseline	1 st
0.8536	0.8360	0.8329	0.7278	0.8547

R3. Gradient boosted regression model (GBR; *estimators* = 1000, *max_depth* = 3) which includes 3 additional features: *w2v_wa*, *ppdb_wa*, *overlap* to feature set used in Run 2.

We used the SVR and LR implementations in Weka 3.6.8. We used the GBR model using sklearn python library. We evaluated our models on training data using 10-fold cross-validation. The correlation scores in the training data were 0.797, 0.816 and 0.845 for R1, R2, and R3, respectively.

3.4.3 Results and Discussions

Table 3.6 presents the correlation (r) of our system outputs with human ratings in the evaluation test data for the STS 2017 English task. The correlation scores of all three runs are 0.83 or above, on par with top performing systems. There were 76 different runs submitted by more than 30 different teams for STS shared task on the English track. All of our systems outperformed the baseline by more than 10 pts. The baseline is the cosine of binary sentence vectors with each dimension representing whether an individual word appears in a sentence. Our R1 system is at par with the 1st ranked system differing by a very small margin of 0.009 (<0.2 pt). The difference is statistically insignificant.

Figure 3.7 presents the graph showing R1 system output against human judgments (gold scores). It shows that our system predicts relatively better for similarity scores between 3 to 5 while the system slightly overshoots the prediction for the gold ratings in the range of 0 to 2. In general, it can be seen that our system works well across all similarity levels.

Feature correlation analysis: We also performed feature correlation

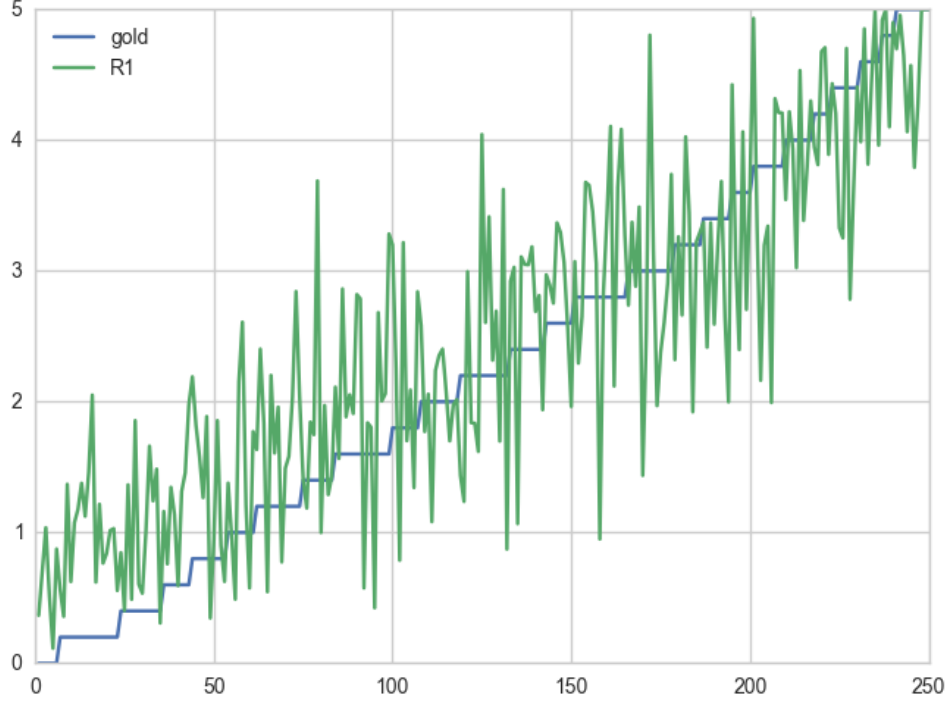


Fig. 3.2: R1 system output in evaluation data plotted against human judgments (in ascending order).

Table 3.7: A set of highly correlated features with gold scores in test data.

<i>dssm</i> (0.8254), <i>ppdb_wa_pen_ua</i> (0.8273),
<i>ppdb_wa</i> (0.8139), <i>cdssm</i> (0.8013),
<i>dssm_lr</i> (0.8135), <i>overlap</i> (0.8048)

analysis with the gold ratings in the test data. Our 11 features correlated 0.75 or above when compared with gold scores in test data. In Table 3.7, we list only those features having correlations of 0.8 or above. Similarity scores computed using word alignment and compositional sentence vector methods were the best predictive features.

Performance against difficult English sentence pairs: Table 3.8 illustrates the difficult English sentence pairs selected by STS 2017 task organizers (Cer et al., 2017). The organizers selected these sentence pairs

Table 3.8: Difficult English sentence pairs (Cer et al., 2017).

Id	Pairs
SP-1	There is a cook preparing food. A cook is making food.
SP-2	The man is in a deserted field. The man is outside in the field.
SP-3	A girl in water without goggles or a swimming cap. A girl in water, with goggles and swimming cap.
SP-4	A man is carrying a canoe with a dog. A dog is carrying a man in a canoe.
SP-5	There is a young girl. There is a young boy with the woman.
SP-6	The kids are at the theater watching a movie. it is picture day for the boys

considering five different challenging issues for semantic similarity: *word sense disambiguation*, *negation*, *compositional meaning*, *semantic blending* and *attribute importance*. For example, they selected first and last sentence pair examples i.e. SP-1 and SP-6 in the Table 3.8 to highlight the issue with the word sense disambiguation, where “*making*” and “*preparing*” have the same meaning in the context of “*food*”, while “*picture*” and “*movie*” are not similar when picture is followed by “*day*”. The sentence pairs SP-3 and SP-4 were selected for negation and compositional meaning aspects, respectively.

Table 3.9 shows the performance of the DT_TEAM system and other top performing systems against these difficult English pair sentences. We notice that our DT_TEAM system also suffers from these issues, particularly negation and semantic blending. The system pumps the semantic score for example sentence pairs SP-3 (negation example) and SP-5 (semantic blending example) by above 1.5 correlation points. However, our system scores, overall, are closer to human scores (correlation = 0.72) as compared against scores assigned by other top performing systems.

Performance on STS benchmark data: We trained our three runs with the benchmark training data under identical settings. We used benchmark

Table 3.9: Performances of top performing STS systems against difficult English sentence pairs (Cer et al., 2017).

Pairs	Human	DT_Team	ECNU	BIT	FCICU	ITNLP-AiKF
SP-1	5.0	4.1	4.1	3.7	3.9	4.5
SP-2	4.0	3.0	3.1	3.6	3.1	2.8
SP-3	3.0	4.8	4.6	4.0	4.7	0.1
SP-4	1.8	3.2	4.7	4.9	5.0	4.6
SP-5	1.0	2.6	3.3	3.9	1.9	3.1
SP-6	0.2	1.0	2.3	2.0	0.8	1.7

Table 3.10: Performances of participating STS systems on STS benchmark data (Cer et al., 2017).

System	Dev	Test
ECNU	84.7	81.0
BIT	82.9	80.9
DT_TEAM	83.0	79.2
UdL	72.4	79.0
HCTI	83.4	18.4
RTM	73.2	70.6
SEF@UHH	61.6	59.2

development data only for generating features using the *tuned sentence representation-based method* (as validation dataset). The correlation scores for $R1$, $R2$ and $R3$ systems were:

In **Dev**: 0.800, 0.822, **0.830** and

In **Test**: 0.755, 0.787, **0.792**

All of our runs outperformed best baseline benchmark system (**Dev** = 0.77, **Test** = 0.72). Interestingly, $R3$ was the best performing while $R1$ was the least performing among the three. The reason might be that $R3$ generalized better with the largest number of features among the three (#features: 7, 8 and 11 for $R1$, $R2$ and $R3$ respectively). Table 3.10 shows the best output of the DT_TEAM system along with the best outputs of other participating systems in STS2017 on STS benchmark data.

3.5 Conclusion

We described our *DT-Team* system that participated in SemEval-2017 Task 1 for English track. We developed three different feature-engineered systems using SVM regression, Linear regression and Gradient Boosted regression models for predicting textual semantic similarity. Overall, the outputs of our models highly correlate (correlation up to 0.85 in STS 2017 test data and up to 0.792 on benchmark data) with human ratings. Indeed, our methods yielded highly competitive results.

However, there is still room for improvement as evidenced by the performance of our system against the difficult English pair examples. In particular, we can explore features that can account for semantics issues such as negation, semantic blending, and compositional meaning aspects.

Chapter 4

Assessing Student Answers in Tutorial Dialogue Context

Automatically assessing open-ended short student responses is an extremely challenging task as students can express their responses in numerous ways owing to different individual styles and varied cognitive abilities and knowledge levels.

Table 4.1 shows four answers, articulated by four different college students, to a question asked by the state-of-the-art intelligent tutoring system (ITS) Deep-Tutor (Rus, DMello, et al., 2013; Rus, Niraula, & Banjade, 2015). It should be noted that all four student answers in Table 4.1 are correct answers to the Tutor Question. As can be seen from the table, some students write full sentences (student answer *A4*), some others write concise answers (*A3*), and yet other students write elaborate answers that include more concepts than are needed (*A1*).

The widely adopted and scalable approach to assessing such open-ended student responses is the semantic similarity in which a score is computed between a target student answer and an expert-provided reference answer (Mohler, Bunescu, & Mihalcea, 2011; Banjade, Maharjan, Niraula, Gautam, et al., 2016). If the student answer has a high semantic similarity score to the reference answer, we infer that the student answer is correct. A low semantic similarity score implies the student response is incorrect. There are two major weaknesses of such semantic similarity approaches. First, contextual information, even when available, is typically ignored. Second, because such approaches rely primarily on the explicit information present in the student response, when student responses vary in the level of explicit information, which is typically the case as shown in Table 4.1, the semantic similarity scores between student response and the reference answer vary widely even within the same category of responses, e.g. correct student responses. We elaborate on these two key observations next.

Table 4.1: An example of a problem and student answers to a tutor question.
Context is needed to assess answers *A1-A3* properly.

Description
Problem description: While speeding up, a large truck pushes a small compact car.
Tutor question: How do the magnitudes of forces they exert on each other compare?
Reference answer: The forces from the truck and car are equal and opposite.
Student answers: <i>A1. The magnitudes of the forces are equal and opposite to each other due to Newtons third law of motion.</i> <i>A2. they are equal and opposite in direction</i> <i>A3. equal and opposite</i> <i>A4. the truck applies an equal and opposite force to the car.</i>

For example, while the student response *A4* can be handled by comparing it to a reference answer, the other student responses in Table 4.1 require more than a reference answer to be properly assessed. For instance, the noun “forces” in student answer *A1* refers to the forces that the truck and the car apply on each other. However, the truck and car need to be inferred from the context as they are not explicitly mentioned in the student response *A1*. A semantic similarity approach that simply compares *A1* with the reference answer would assign a low similarity score to this student response because important concepts in the reference answer (truck and car) are missing, i.e., not explicitly mentioned in the student response *A1*.

The role of context to correctly interpret a short student response in tutorial dialogues is also illustrated by answer *A2* in Table 4.1. In this case, the pronoun “*they*” is referring to the amounts of forces exerted by the car and truck on each other. Pronouns, such as *they*, *he*, *she*, and *it*, are frequently employed by students in tutorial dialogues. A key finding that informed our work was reported by Niraula et al. (2014), who analyzed conversational tutorial logs, and showed that 68% of the pronouns used by students were referring to entities in the previous dialogue turn or

the problem description. Based on this finding, we consider as context in our work only the problem description and the previous tutor utterance as opposed to the full tutorial dialogue history, which would make the task computationally even harder.

Answer *A3* in Table 4.1 is also challenging to assess with a semantic similarity approach that only considers the reference answer. The challenge spawns from the fact that *A3* is elliptical, i.e., it is in the form of an incomplete sentence whose missing parts are implied by the context. Without considering context, a typical semantic similarity approach would assign a low similarity score to such elliptical responses even if they are correct responses.

Though there have been many approaches proposed to the task of automatically assessing freely generated student answers (see Section 4.1), interestingly there are not many works that consider context when evaluating such student answers.

One approach to account for context when assessing student answers is to consider only new content/concepts in student answers that are not present/mentioned in the previous context similar to Bailey and Meurers (2008). That is, if the student response repeats the concepts mentioned in the question, no new information is provided. Such concepts are ignored in their approach. For example, the concepts of force *from the truck* and *force from the car* are less important when assessing the student answers in Table 4.1 because these concepts are mentioned in the previous context, i.e., question and the problem description. Pronouns in student answers usually co-refer to words in context and therefore can be considered as given information when assessing student answers. In conclusion, this Bailey and Meurers (2008) approach can implicitly handle, by simply ignoring, both co-referring pronouns and ellipses (see. *A2*, *A3* in Table 4.1) as given concepts implied by context are ignored.

Nevertheless, given concepts in student answers can be useful as they

indicate the relevancy of the student answers to the question and the broader context, which also includes the problem description. The better approach, as argued by Banjade, Maharjan, Niraula, Gautam, et al. (2016), is to assign less weight (instead of completely ignoring them by assigning a zero weight) to given concepts in the student answers. This would provide non-zero scores to heavily context-dependent answers while still giving lower scores to responses having fewer new concepts compared to responses rich in new concepts. In one of our approaches, we apply this basic idea to compute word weighted lexical and alignment similarity scores and used them as features in addition to various context aware count features in a Gaussian Mixture Model (McLachlan & Peel, 2004) model.

In the second approach, we present an LSTM-based network approach for handling student answers in context. Specifically, we handle context inherently by utilizing LSTM layers (Hochreiter & Schmidhuber, 1997) to capture long-term dependencies between a target student answer to be assessed and the previous context in which that answer was generated by the student.

4.1 Related Work

The WebLAS system (Bachman et al., 2002) assessed short answer for questions in a language learning task by matching student answers against regular expressions of the corresponding reference answers. The C-Rater (Leacock & Chodorow, 2003) system normalized concepts in student answers by handling various linguistic variations and then assessed the student answers by looking at the number of concepts that match concepts within the reference answer. The Intelligent Assessment Technologies system (Mitchell, Russell, Broomhead, & Aldridge, 2002) and Oxford system (Pulman & Sukkarieh, 2005) matched manually created information extraction templates corresponding to questions against student responses. The Atenea system (Pérez et al., 2005) used Latent Semantic Analysis (Landauer et al., 1998) and n-gram overlap for assessment.

The Content Assessment Module (Bailey & Meurers, 2008) system aligned concepts in student responses with concepts in the target responses at token, chunk and relation levels. Finally, the aligned and unaligned concepts were used to develop a classifier to evaluate the student answers. Nielsen, Ward, and Martin (2009) mapped reference responses and student responses into meaningful parts called facets. The facets were then aligned and annotated with five different semantic labels and used with a machine learning approach to make answer assessment decisions. Mohler et al. (2011) aligned student answers and reference answers using subgraphs of the dependency structures for answer assessment. Most of the works presented above, including the recent Semantic Textual Similarity (STS) shared tasks (Agirre et al., 2015; Agirre, Banea, et al., 2016; Cer et al., 2017), focus on comparing student answers against reference answers in isolation. They do not consider the broader context (e.g., question, problem description) in which the student response was generated. One of just a few previous works that used context in their assessment is an approach by Bailey and Meurers (2008), as already mentioned. They used question text as the context for implicit pronoun resolution and distinguished between new concepts and given concepts. They assessed student answers against target answers based on the alignment of new concepts alone. Given concepts were discarded, as explained earlier. In our approach, we do not discard the given concepts.

One of the reasons that explains the lack of research on assessing short answers in context may also be attributed to a lack of appropriately annotated datasets that consider contextual information. Banjade, Maharjan, Niraula, Gautam, et al. (2016) recently released the DT-Grade dataset that was annotated with contextual information. They defined four qualitative levels of correctness: *Correct*, *Correct-but-incomplete*, *Contradictory*, and *Incorrect*. They also presented a baseline model for assessing students answers that accounts for context by giving a

non-zero, albeit small, weight to given words, i.e., words in the student answer that are mentioned in the previous utterance. New words in the student answer are given a large weight when computing the similarity score between the student answer and the reference answer. Their baseline model used a single similarity feature (F7 in Table 4.2) and built a logistic classifier to predict the correctness label.

In our work, we used a variety of similarity features computed using a context-based word weighting approach. We also used features that rely on context-based counts in conjunction with the GMM model to perform soft clustering that can account for variability in the student answers for each of the four answer correctness levels in the DT-Grade dataset. GMM is also useful to understand how different features are interacting and contributing to predicting student answer correctness labels.

Next, we developed an LSTM approach to assess free short answers in tutorial dialogue contexts. In the LSTM approach, we handled context inherently by utilizing LSTM layers (Hochreiter & Schmidhuber, 1997) to capture long-term dependencies between a target student answer to be assessed and the previous context in which the student-generated that answer. A major advantage of the LSTM method is that it does not require any feature engineering, and it performs on par and even slightly better than the GMM method.

4.2 Approach

We present two different approaches to assessing student answers in context below.

4.2.1 Gaussian Mixture Model

Our approach using the Gaussian Mixture Model is based on the observation that similarity scores between student answers and the reference answers vary greatly for each of the four correctness levels in the DT-Grade dataset: *Correct*, *Correct-but-incomplete*, *Contradictory*, and *Incorrect*. To illustrate our point, we focus on the correct student answers shown in Table 4.1.

The number of new and given concepts vary across the four student answers in Table 4.1, as explained next. We define new and given concepts relative to the reference answer and the given context (the question and problem description). The number of new and given concepts for the complete answer *A4*, the elliptical answer *A3*, the pronoun referring answer *A2* and the answer containing extra information *A1* are: (3 new, 3 given), (2, 0), (3, 0) and (6, 2). This wide range of the number of new and given concepts for the correct answers also holds for student answers marked with other correctness labels, e.g., contradictory. The semantic similarity scores for student answers also vary greatly for each of the correctness labels due to the diversity of student answering “*styles*”, as explained earlier: some students rely heavily on context and provide short responses while others write well-formed full sentences. Further, while incorrect student answers in the DT-Grade dataset usually have low similarity scores against reference answers, in some cases they have high semantic similarity scores. For instance, the reference answer “*Newton’s first law of motion*” and the corresponding incorrect student answer “*Newtons second law of motion*” have a high similarity score. Therefore, we modeled the correctness levels as multivariate Gaussian densities of feature vectors, where features are different counts of new and given concepts and sentence similarity scores (see Section 4.2.1 for detail).

Our Gaussian Mixture Model is a probabilistic model represented as a weighted sum of M mixture components in the following equation.

$$p(x|\lambda) = \sum_{i=1}^M w_i N(x|\mu_i, \Sigma_i) \quad (4.1)$$

where x is a d -dimensional continuous feature vector, w_i are mixture component weights, and $N(x|\mu_i, \Sigma_i)$ are multivariate normal distributions

Table 4.2: Features used in model development.

FN	Features
Count (CNT) Features	
F1	Unique content word count in reference answer
F2	Unique content word count in student answer
F31	Percentage of Unique content word count in reference answer
F4	Percentage of Unique content word count in student answer
F5	Common content word count in reference answer
F6	Common content word count in student answer
Alignment (ALGN) Features	
F7	word2vec (Mikolov et al., 2013) similarity score
F8	GloVe (Pennington et al., 2014) similarity score
F9	LSA wiki (Stefănescu et al., 2014) similarity score
Lexical (LEX) Features	
F10	Lexical Overlap similarity score
F11	Vector Cosine similarity score

representing the component Gaussian densities with mean vector μ_i and covariance matrix Σ_i for $i = 1, \dots, M$.

In our GMM model, we model the four correctness levels as multivariate Gaussian densities of feature vectors. The feature vector contains the features described next.

Features: Our context-based features can be grouped into three groups, Count Features, Alignment Features, and Lexical Features.

Count Features: We consider only content words for the count features. We adopt a more meaningful terminology and refer to new content words in student/reference responses as “*unique*” and to content words that were provided/given in the previous context as “*common*” (they are common in the student/reference answer and the context). We generated six count features, F1-F6 in Table 4.2.

Alignment Features: We computed alignment-based similarity scores (F7-9 in Table 4.2) using Equation 4.2 based on a word weighting approach that accounts for context as described in (Banjade, Maharjan, Niraula, Gautam, et al.,

2016). The “*unique*” words in student answers and reference answers are given full weight (weight=1.0). “*Common*” words are given less weights compared to “*unique*” words.

$$sim(A, R) = 2 * \frac{\sum_{(a,r) \in OA} w_a * w_r * sim(a, r)}{\sum_{a \in A} w_a + \sum_{r \in R} w_r} \quad (4.2)$$

where OA is optimal alignment of words between A and R obtained using Hungarian algorithm as described in Rus and Lintean (2012). A/R refers to the student/reference answer, respectively, and a/r is a token in the corresponding answer. $Sim(a, r)$ is the cosine similarity score between A and R . The weights $0 \leq w_a \leq 1$ and $0 \leq w_r \leq 1$ refer to the weight of a word in A and R , respectively. We empirically set the word weights w_a and w_r as described in the *Experiments and Results* section.

A **lexical overlap score** (F10 in Table 4.2) was computed as the percentage of word overlap in the student and reference answers (Equation 4.3). WordNet was employed to normalize variation due to synonyms.

$$sim(A, R) = 2 * \frac{\sum_{a \in A, r \in R} w_a * w_r * sim(a, r)}{\sum_{a \in A} w_a + \sum_{r \in R} w_r} \quad (4.3)$$

A **vector cosine score** (F11) was computed as cosine similarity between vector representations of the student and reference answers. We represented the student and reference answers as vectors: $V_A = (w_{1a}, w_{2a}, w_{3a} \dots w_{na})$ and $V_R = (w_{1r}, w_{2r}, w_{3r} \dots w_{nr})$, where each $word_i$ in vocabulary voc represents a dimension of the vector. The voc is constructed from the unique words in student and reference answers. V_A/V_R vector is generated by empirically setting the value of

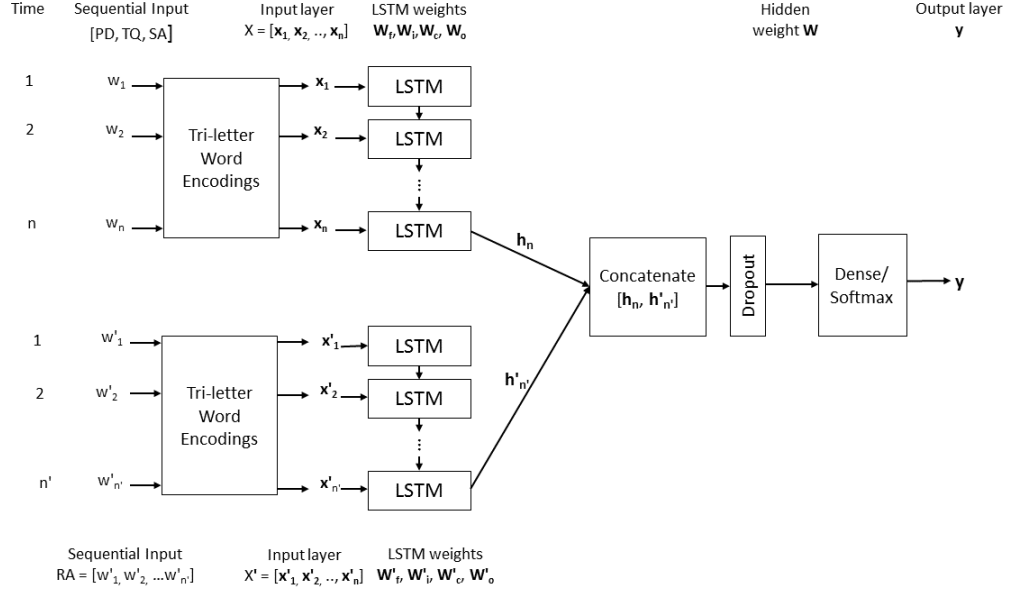


Fig. 4.1: LSTM model architecture with tri-letter word encodings.

the weights $w_{ia/r}$ from 0 to 1.

$$w_{ia/r} = \begin{cases} 0 & \text{if } word_i \text{ is not in A/R,} \\ 1 & \text{if } word_i \text{ is in A/R but not in context,} \\ w_{ia/r} & \text{if } word_i \text{ is in both A/R and context} \end{cases} \quad (4.4)$$

4.2.2 The LSTM Approach

Figure 4.1 illustrates our approach based on an LSTM network architecture. To describe the network, we use lowercase bold letters such as \mathbf{x} to denote column vectors and uppercase letters such as \mathbf{W} to denote matrices.

We generate a tri-letter vocabulary (size 1,147) from all the words in the DT-Grade dataset (see Section 4.3.3) and use the vocabulary to encode the words as vectors (Huang et al., 2013). For example, we padded words such as “car” on both sides with a word boundary symbol “#” resulting in the following token “#car#”

from which we then generated the following tri-letters “#ca”, “car” and “ar#”. We generated a tri-letter vocabulary of size 1,147.

The input consists of the extended student answer which is the result of concatenating the previous context, i.e., the related problem description and the previous tutor question, to the student answer. For example, an extended answer based on the student answer A1 in Table 4.1 will be [PD, TQ, A1] : “*While speeding up, a large truck pushes a small compact car. How do the magnitudes of forces they exert on each other compare? They are equal and opposite in direction*”. The second input is the expert provided reference answer RA: “*The forces from the truck and car are equal and opposite.*”. The input is a sequence of words which are represented as word vectors. In both figures, $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ and $X' = [\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_{n'}]$ represent the sequential inputs [PD,TQ,SA] and RA, respectively, where \mathbf{x}_i and \mathbf{x}'_i indicate the word vector representations for words w_i and w'_i , respectively. n and n' refer to the length of the sequence X and X' . It should be noted that not extending the expert provided reference answers with contextual information was a conscious design decision - these answers are typically self-contained, i.e. they do not require an expansion to recover implied information. Further, the network consists of two LSTM components corresponding to the two sequential inputs. The first LSTM component generates an embedded representation output \mathbf{h}_n at time step n (end of sequence X) while the other LSTM component generates an embedded representation $\mathbf{h}'_{n'}$ at time step n' (end of sequence X'). We generated an embedded representation of both the extended student answer and the expert provided reference answer $[\mathbf{h}_n, \mathbf{h}'_{n'}]$ by concatenating both LSTM outputs. This embedding is then fed to a dense softmax layer via a dropout layer to produce an output classification vector \mathbf{y} which represents the correctness class for the given student answer instance. A dropout is a popular technique to regularize deep networks (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) where

neurons, i.e., network cell units, are randomly dropped during network training.

The student answers in our case can fall into one of the following correctness classes: *correct*, *partially correct*, *incorrect* or *contradictory*.

4.3 Experiments and Results

We experimented with both GMM and LSTM approach and evaluated their performance using the DT-Grade dataset. If there were more than one reference answers to a question, we chose the one with the highest similarity score to the target student answer.

4.3.1 Data

DT-Grade (Banjade, Maharjan, Niraula, Gautam, et al., 2016) consists of 900 student responses gathered from an experiment with the DeepTutor intelligent tutoring system (Rus, DMello, et al., 2013; Rus, Niraula, & Banjade, 2015). The student answers were extracted from logged tutorial interactions between 40 junior level college students and the DeepTutor system while solving conceptual physics problems. The student answers are annotated with one of the following correctness labels.

- *Correct (C)*: The student answer is correct in context. Extra information, if any, in the answer is not contradicting with the reference answer.
- *Correct-but-incomplete (CBI)*: The student answer covers the expected concepts in the reference answer only partially and contains no incorrect parts.
- *Contradictory (CNTR)*: Student answer is opposite or contrasting to reference answer. For example, “*equal*”, “*less*”, and “*greater*” are contradictory to each other.
- *Incorrect (IC)*: If none of the above labels apply, the student answer is deemed incorrect. *Contradictory* is a subset of *Incorrect* category

Table 4.3: Performances of different GMM models. Baseline model M0 is a logistic model presented by Banjade, Maharjan, Niraula, Gautam, et al. (2016). U-CNT refers to Unique Counts. *M14** model was developed using instances not requiring contextual information alone and evaluated on instances requiring contextual information. *LSTM** refers to our LSTM approach (Maharjan et al., 2018).

Model	Features	F	A(%)	K
M0	Baseline	-	49.33	0.22
M1	CNT only (F1-6)	0.433	44.40	0.21
M2	ALGN only (F7-9)	0.454	52.10	0.27
M3	LEX only(F10-11)	0.421	51.20	0.25
M4	CNT + ALGN (F1-6, F7-9)	0.555	55.60	0.37
M5	CNT + LEX (F1-6, F10-11)	0.545	54.40	0.35
M6	ALGN + LEX (F7-11)	0.498	51.90	0.30
M7	All features (F1-11)	0.560	55.80	0.37
M8	M1 + M2 + M3 ensemble	0.506	54.60	0.31
M9	U-CNTs only (F1-4)	0.401	45.40	0.20
M10	U-CNT + LEX (F1-4, F10-11)	0.563	56.80	0.37
M11	U-CNT + ALGN (F1-4, F7-9)	0.568	57.4	0.39
M12	All excluding common Counts (F1-4, F7-11)	0.58	58.2	0.40
M13	M9 + M2 + M3 ensemble	0.500	55.3	0.32
<i>M14*</i>	Same as M12	0.545	54.8	0.34
<i>LSTM*</i>	-	0.62	62.22	0.45

In addition to the annotation for correctness, the student answers were also annotated for: (a) whether contextual information was helpful to interpret the student answer correctly, and (b) whether the student response contains extra information, i.e., not mentioned in the ideal answer. The DT-Grade dataset contains about 25% of student answers that require contextual information to assess them properly.

4.3.2 GMM based Experiments and Results

We developed different GMM models with different combinations of features. We used the Expectation Maximization (EM) algorithm (Dempster et al., 1977) for estimating the GMM parameters. We followed a 10-fold cross validation methodology for model evaluation except for M14 model. We report F-measure (F), Accuracy (A), and Kappa (K) as performance metrics.

Context based word weighting: Both context based lexical and alignment methods rely on word weights, which need to be inferred, to compute similarity scores. If we give a full weight of 1 to each word in the student answer and the reference answer, we assume both unique and common (i.e., also present in context) words are equally important. If we set a weight of 0, say for the common/given words, we are completely discarding the common/given words. To find the best word weights, a grid search process was run in which we computed similarity scores by varying the weight value from 0 to 1 in increments of 0.1. We then created different logistic classifiers using each of the similarity scores alone and observed the classification accuracies while changing the weights. All the alignment based and lexical based similarity features (F7-11) performed best or close when we set the word weight to 0.4. Therefore, we computed these scores with the word weight set to 0.4 when using them as features for GMM models.

Table 4.3 shows that model M2 using alignment-features outperformed the count-features only model M1 and the lexical-overlap-feature model M3 across all performance metrics. The M2 model also outperformed the baseline model (Banjade, Maharjan, Niraula, Gautam, et al., 2016) which uses an alignment score computed using the word2vec word representation (Mikolov et al., 2013).

If we look at the combination of feature groups, the count features are significantly complimenting lexical or alignment features. Though the improvement in terms of accuracy of models M4/M5 over M2 model was a 2-3%, the gain in terms of the F-measure was about 0.1 (10% increase), which is significant. The all-feature model M7 was the best performing model across all three performance metrics while the ensemble classifier M8 outperformed the basic classifiers M1, M2 and M3, but lagged behind the all-feature model M7.

The common content features are implicitly captured by the unique content word proportion features ($\text{correlation}(F3, F5) = -0.51$, $\text{correlation}(F4, F6) =$

-0.58). Therefore, we also explored different combinations of features excluding the common count features. Interestingly, all types of combinations consistently outperformed their counterparts that included common count features (see M10 vs M5, M11 vs M4, M12 vs M7 and M13 vs M8 in Table 4.3).

The best of all was the M12 model which included all the features except the common count features with an F-measure of 0.58 and accuracy of 58.2% ($\kappa = 0.4$). Also, the performance of the M14 model built using data not requiring contexts and tested on the 25% instances that require contextual information is comparable to M12 performance.

Interpreting the results based on GMM feature analysis: We analyze how various features contribute to the predictive power of our approach. This information can be helpful in interpreting the output and in finding irrelevant features which do not contribute significantly to the overall performance of our approach. We can easily perform such a feature analysis with a probabilistic GMM model by computing the membership weights for each of the student answer reference answer pairs/instances with respect to each of the four correctness classes/levels using the all-feature M7 model. Then, we perform a Pearson correlation analysis to find the features that highly correlate with the GMM weights for each of the four correctness levels as shown in Table 4.4.

Table 4.4 provides interesting insights. First, it supports our intuitions related to answers belonging to the different correctness labels. For example, the similarity features (F7-11) highly positively correlate with *Correct* labels and highly negatively correlate with *Incorrect* labels. Also, the negative correlation between unique count features (F1-4) and the *Contradictory* label weights indicates that instances annotated contradictory have relatively fewer *unique* content words in the student and reference answers.

Table 4.4 also indicates that students are more likely to provide more new

Table 4.4: Correlation analysis between various features and GMM weights for correctness labels (CNT - Contradictory, C - Correct, CBI - Correct-but-incomplete, IC - Incorrect) computed using model M7. F1-F11 are the features from Table 4.2. Moderate or higher correlation scores are marked as bold.

	CNTR	C	CBI	IC
F1	-0.265	-0.222	0.539	-0.028
F2	-0.322	0.240	0.033	-0.13
F3	-0.306	-0.096	0.419	-0.056
F4	-0.441	0.213	0.007	0.0007
F5	0.049	-0.161	0.046	0.131
F6	0.14	-0.031	-0.024	-0.026
F7	-0.298	0.700	-0.030	-0.665
F8	-0.318	0.635	-0.009	-0.590
F9	-0.357	0.637	-0.005	-0.572
F10	-0.073	0.541	0.142	-0.758
F11	-0.196	0.513	0.090	-0.600

content when they are correct than incorrect (see the correlation between F2 and F4). Similarly, the high correlation for the unique content words in reference answers (F1 and F3) indicates that the instances where there are a higher number of unique counts in the reference answer, the students are more likely to provide partially correct answers. Intuitively, this makes sense as the presence of more unique words in reference answers indicates the question is asking the student to cover many concepts in his response.

Second, the feature analysis in Table 4.4 provides insights and explains the performance of the different GMM models. The alignment features (F7-9) correlate reasonably or highly for more correctness levels than it is the case for the count features (F1-6) and lexical features (F10-11). This supports the fact that Model M2 is performing better than models M1 and M3. Another interesting observation is the fact that lexical and alignment features are useful for predicting if the student answers are contradictory, correct or incorrect. They are not as good at predicting partially correct answers. On the other hand, the count features contribute towards predicting all correctness labels except the *Incorrect* label. This can explain why the

Table 4.5: Performance of The LSTM Models. Accuracy values are in percentage followed by Kappa values in brackets.

Model	F-score	Accuracy
Logistic Model (Banjade et al., 2016)	-	49.33(0.22)
GMM Model (Maharjan, Banjade, & Rus, 2017)	0.58	58.2(0.40)
NN_1 (50 LSTM cells, Tri-letter word encodings)	0.53	53.3(0.33)
NN_2 (100 LSTM cells, Tri-letter word encodings)	0.55	55.4(0.36)
NN_3 (50 LSTM cells, Pretrained GloVe embeddings)	0.6	60.11(0.42)
NN_4 (100 LSTM cells, Pretrained GloVe embeddings)	0.62	62.22(0.45)

combination of count features with lexical/alignment features worked better than combining lexical with alignment features (see M4/M5 vs. M6 in Table 4.3).

4.3.3 LSTM based Experiments and Results

We experimented with four different types of neural networks by varying the number of LSTM cell units and type of word vector representation (see Table 4.5). We used a batch (number of samples per gradient update) of size 30 and a dropout rate of 0.5 (fraction of the input units to drop) empirically. We used a 10-fold cross validation methodology for evaluating trained networks.

The results shown in Table 4.5 indicate that increasing the number of LSTM cell units while keeping other factors constant improved the performance of the underlying model, in general (NN_1 vs NN_2, NN_3 vs NN_4). The reason might be that an increased number of LSTM cell units yielded better embedded representations for the sequential inputs. However, determining the optimal number of LSTM cell units is an open question and therefore selecting the number of LSTM cell units is mostly empirically driven. Next, we found that using pre-trained GloVe embeddings over tri-letter encodings improved the network performance (NN_1 vs NN_3, NN_2 vs NN_4). This result suggests that using pre-trained word embeddings such as GloVe to represent words is useful. Such word embeddings are typically developed from a large corpus of text and therefore can better represent word semantics than simple word-based or tri-letter based one-hot encodings.

Furthermore, using pre-trained models fits the now well-adopted approach to training deep neural network architectures: pre-training. That is, pre-training is used as a successful way to avoid the vanishing gradient problem (the gradients become smaller and smaller as they are back-propagated which prevents the weights of the earlier layers from updating during training) and improve the performance of deep neural networks.

We also note that all our neural networks performed better than a logistic model based on a single sentence similarity computed using a word alignment method which weights words based on context (Banjade, Niraula, et al., 2015). None of our networks using tri-letter word encodings outperformed the GMM based method (Maharjan, Banjade, & Rus, 2017). On the other hand, NN_3 and NN_4 with pre-trained GloVe word embeddings did outperform the state-of-the-art methods on the DTGrade dataset. The best model NN_4 yielded an average F1-score of 0.62 and an average accuracy of 62.2% and a kappa of 0.45. A larger training dataset may lead to better results for both the tri-letter models and the GloVe models.

4.4 Conclusion

We presented a probabilistic Gaussian Mixture model using multiple context-aware features based on counts, and word weighted lexical and alignment similarity scores to assess the student answers in context and general. Our best performing model achieved significant improvement of 9% in terms of accuracy and almost twice the kappa value over a baseline system. Additionally, we showed how the GMM model might help to understand how different features are contributing to the overall answer assessment performance.

Next, we presented a novel deep learning approach using LSTM to assess free student answers in tutorial dialogues by taking context into account. The approach outperforms state-of-the-art methods evaluated on the DTGrade dataset. Our

approach is particularly useful because student answers can vary significantly in the level of explicit information they contain. Additionally, it does not require the tedious task of manually crafting features.

There is room for improvement of our approaches. We can explore more distinctive context-based features for our GMM method. Also, we can further our investigation of using deep neural networks for assessing students' freely generated responses. Eventually, we plan to combine the proposed solutions with an alignment-based solution such that besides a holistic outcome such as *Incorrect* or *Contradictory* we also generate an explanation for the decision which would enable dynamic and automatic generation of personalized hints.

Chapter 5

SemAligner: A Tool for Interpreting Semantic Textual Similarity

The Semantic Textual Similarity (STS) competitions have been held since 2012 to improve methods for measuring semantic textual similarity (Agirre et al., 2012, 2013, 2014, 2015; Agirre, Banea, et al., 2016; Cer et al., 2017). As such, the STS systems have greatly improved in recent years on predicting similarity score for a given pair of short texts. The state-of-the-art STS systems (Maharjan, Banjade, Gautam, et al., 2017; Tian et al., 2017) give correlation above 0.85 on STS 2017 evaluation data while they give correlation above 0.8 on the STS benchmark data (Cer et al., 2017).

However, these systems do not explain why the two texts are similar, related or unrelated. For example, consider a question asked by DeepTutor, its expected answer and one of the student responses to the question asked by the system as shown in the Table 5.1.

Any top performing STS system would give a similarity score of 3 meaning that the student answer is missing some important information. However, it does not explain which information is missing. If there existed explanatory functionality that could explain the student is missing information on *direction*, we could use this diagnostic information to generate a follow-up question by the tutor as: *What other information than magnitude is provided by the acceleration?* Such explanatory layer would make a big difference in many Natural Language Processing (NLP) applications such as intelligent tutoring system (Graesser et al., 2005; Rus, Niraula, & Banjade, 2015) and student answer evaluation (Rus & Graesser, 2006; Nielsen et al., 2009).

One approach towards providing an explanatory layer to STS systems is to align the chunks in a given pair of texts and label them with semantic relation types

Table 5.1: A sample question and answer between student and DeepTutor with the ideal expected answer.

Description
Question: Because it is a vector, acceleration provides what two types of information?
Student Answer: Acceleration gives magnitude
Expected Answer: Acceleration provides magnitude and direction.

and scores as proposed in the pilot interpretable Semantic Textual Similarity (iSTS) task (Agirre et al., 2015). A chunk is a syntactically meaningful unit which typically consists of a single content word surrounded by a group of function words (Abney, 1991). Agirre et al. (2015) introduced a set of semantic relation types: EQUI (chunks are semantically equivalent), OPPO (chunks are opposite in meaning), SPE1/SPE2 (the chunk in the first/second sentence is more specific than the chunk in the second/first sentence), SIMI (chunks are similar but not EQUI, OPPO or SPE), REL (chunks are related but not EQUI, OPPO, SPE or SIMI), ALIC (a chunk is not aligned to any other chunk due to 1:1 alignment restriction) and NOALIC (the chunk is unrelated and has no alignment).

The iSTS tasks were organized in 2015 and 2016 (Agirre et al., 2015; Agirre, Gonzalez-Agirre, et al., 2016) to foster research towards improving interpretable STS systems. We participated in the tasks in both years. Our interpretable systems (Banjade, Niraula, et al., 2015; Banjade, Maharjan, Niraula, & Rus, 2016) were top performing systems. The interpretable DTSim system (Banjade, Maharjan, Niraula, & Rus, 2016) was a slightly improved version of our earlier system NeRoSim (Banjade, Niraula, et al., 2015) since the system supported many to many alignments. The interpretable DTSim system exploited the functionality of the SemAligner tool (Maharjan et al., 2016) to handle both chunked or plain text-pairs as input.

We describe our SemAligner tool in this chapter which can align textual

chunks in a given pair of texts (chunked or plain texts). The SemAligner also assigns semantic relation types and semantic similarity scores to the aligned chunks; thus, the proposed SemAligner tool creates a new category of natural language processing tools called semantic aligners.

We originally developed SemAligner based on the NeRoSim system (Banjade, Niraula, et al., 2015). However, NeRoSim system can handle only gold chunks of the given text-pairs; the task organizers provided these chunks. In addition to the chunked texts, the SemAligner tool can handle plain texts by automatically chunking them using two powerful automated chunkers, namely Extended Open-NLP (EONLP) chunker and CRF-based chunker (Section 5.2.1). Our experiments, described later, show that the performance of the tool in both system-generated and gold chunk categories is better or competitive to other systems.

5.1 Related Work

Most semantic similarity methods are geared towards quantifying the similarity between a pair of texts. Works towards interpreting similarity, i.e., justifying why the two texts are similar or dissimilar, are limited but gaining momentum as described next.

Brockett (2007) annotated datasets to indicate alignment of words and phrases. Other related works are word or phrase-based alignment models for statistical machine translation (Och & Ney, 2004) and word alignment tools. A most recently released tool is the monolingual word-aligner (Sultan, Bethard, & Sumner, 2014) which works at word level but lacks capabilities to assign semantic relation types. In the area of student answer assessment, Nielsen et al. (2009) aligned facets/words in student response with concepts in the reference answer for textual entailment. All these previous works focused primarily on the alignment task without attempting to label the semantic relations among the aligned tokens.

The first attempt to assign semantic labels to aligned tokens is by Rus et al. (2012) who aligned words using greedy and optimal strategies and presented a method to annotate texts with semantic relations such as IDENTICAL and RELATED at the word level. More recently, the already mentioned iSTS tasks in SemEval 2015 and SemEval 2016 (Agirre et al., 2015; Agirre, Gonzalez-Agirre, et al., 2016) focused on labeling aligned chunks with different semantic relation types and semantic similarity scores thereby providing an explanatory layer to the core semantic similarity task. Our SemAligner tool makes contributions toward the development of such powerful, interpretable STS and other NLP systems.

5.2 The SemAligner tool

Figure 5.1 shows the system pipeline of the SemAligner tool. The tool can take chunked or plain text-pairs as input. If the input text-pairs are in plain text format, the tool runs the Chunker component to identify and create chunks. The user can configure the choice of the chunker, either EO-NLP or CRF chunker (Section 5.2.1), in the application configuration file. It should be noted that before performing chunk alignment, the SemAligner preprocesses the text-pairs by performing stopword marking (stopwords are marked to differentiate them from content-words; some rules use this information), lemmatization, POS tagging and Named-Entity recognition using the Stanford CoreNLP Toolkit (Manning et al., 2014).

5.2.1 Chunker

We developed a Conditional Random Field (CRF)¹ based chunker using both CoNLL-2000² shared task training and test data. This data consists of Wall Street Journal corpus: sections 15-18 as training data (211,727 tokens) and section 20 as test data (47,377 tokens). A CRF is a probabilistic graphical model which is often used for labeling sequential data. For chunking task, we generated features such as previous and next words from the current word, current word itself, current word

¹<https://taku910.github.io/crfpp/>

²<https://www.clips.uantwerpen.be/conll2000/chunking/>

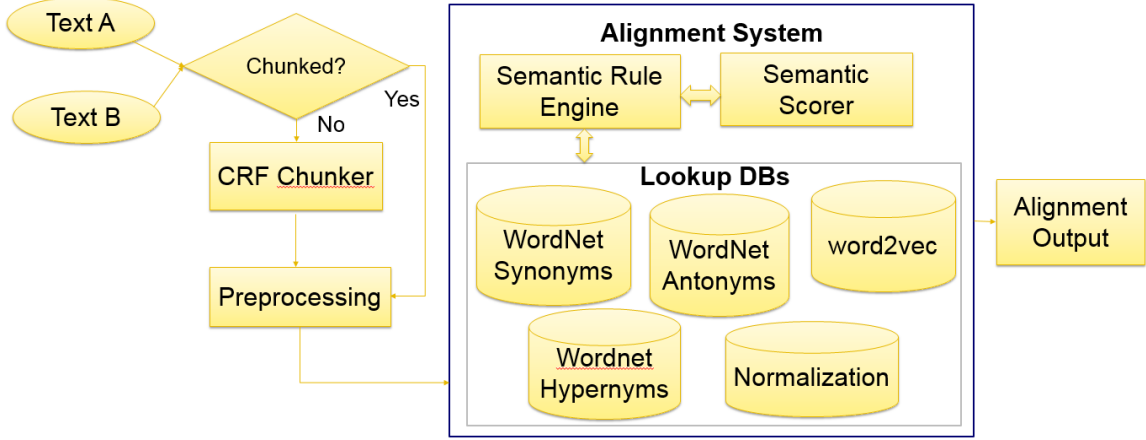


Fig. 5.1: System pipeline of SemAligner tool.

POS tag, previous and next word POS tags and their different combinations as described in Sha and Pereira (2003) for building the CRF model.

Table 5.3 illustrates the gold chunks and chunks generated by our CRF chunker for a sentence. We evaluated the chunking accuracy of the CRF chunker by comparing it against the gold chunks of iSTS 2015 data: the training and test data set each consist of 375 pairs of Images annotation data and 378 pairs of Headlines texts. This chunker yielded the highest average accuracies on both the training and test datasets compared to other chunkers (Table 5.2). The accuracies on the training dataset were 86.20% and 68.34% at chunk and sentence level, respectively. For the test dataset, the accuracies were 86.81% and 69% at chunk and sentence level, respectively. We also chunked the input texts using the Open-NLP³ chunking library (O-NLP). The results are presented in Table 5.2. The average (of Images and Headlines data) accuracies were 53.04% at chunk level and a modest 9.27% at sentence level for the training dataset. It yielded similar results on test data.

Given the modest performance of the O-NLP chunker, we analyzed its output (i.e. chunks) and added the following rules to merge some of the chunks which resulted in chunks that make more sense and led to significantly better performance.

³<http://opennlp.apache.org/cgi-bin/download.cgi>

Table 5.2: Comparison of chunking accuracies of the various chunkers at chunk level (CL) and sentence level (SL) using gold chunks from the iSTS task 2015 data.

DataSet	Chunker	CL (%)	SL (%)
Training Data			
Headlines	O-NLP	53.74	13.49
	EO-NLP	80.67	59.39
	CRF	82.60	62.56
Images	O-NLP	52.35	5.06
	EO-NLP	89.13	72.66
	CRF	89.74	74.13
Test Data			
Headlines	O-NLP	53.88	16.13
	EO-NLP	80.96	60.18
	CRF	83.32	63.23
Images	O-NLP	52.71	5.33
	EO-NLP	89.30	72.13
	CRF	90.29	74.93

Table 5.3: A sentence from Headlines training data chunked by our CRF chunker.

Description
Text: China stocks close higher after economic meeting
Gold Chunk: [China stocks] [close] [higher] [after economic meeting]
Predicted: [China stocks] [close higher] [after economic meeting]

$$(a) PP + NP \Rightarrow PP$$

$$(b) VP + PRT \Rightarrow VP$$

$$(c) NP + CC + NP \Rightarrow NP$$

For example, EO-NLP chunker merges chunks *[on]* and *[Friday]* to form single PP chunk *[on Friday]* using rule (a). The Extended Open-NLP chunker (EO-NLP) reported 84.9% chunk level and 66.02% sentence level accuracies, respectively, on average on the training dataset. The accuracy of the test data was comparable at 85.13% chunk level and 66.15% sentence level.

Both the EO-NLP and CRF chunkers are available as part of the SemAligner tool.

5.2.2 Alignment System

Once the chunks are preprocessed, the SemAligner runs *Semantic Rule Engine* (a set of rules) to align chunks and detect the semantic relation labels. We discuss the rules only briefly here since they are explained in detail in Banjade, Niraula, et al. (2015). There is a subset of alignment rules for each semantic relation type. There are 5 EQUI rules, 1 OPPO rule, 3 SPE rules, 5 SIMI rules, 1 ALIC rule, and 1 NOALIC rule. The rules are applied only when certain conditions are met. While aligning chunks, these rules are applied in the following order of precedence: NOALIC, EQUI, OPPO, SPE, SIMI, REL, and ALIC. Also, there is a precedence of rules within each relation type. For example, the rule *Both chunks have same tokens* (E.g. *to compete* \Leftrightarrow *To Compete*) is always applied first before other EQUI rules.

Our SemAligner tool relies on synonym, antonym and hypernym relations to align the chunks and therefore use several lookup files to determine these word-to-word semantic relations. All these lookup resources were created using WordNet (Miller, 1995). There are also rules that use the similarity score between two chunks for determining the alignment. Word to word similarity measures are used to measure chunk to chunk similarity using optimal alignment as described in Ștefănescu et al. (2014). Currently, we use cosine of vectors using the word2vec model (Mikolov et al., 2013) as the word-to-word similarity measure as illustrated by the following rule, *if Both chunks have an equal number of content words and $\text{sim-Mikolov}(C1, C2) > 0.6$, label as EQUI*. The similarity threshold 0.6 was selected empirically after trying with thresholds varying from 0.4 to 0.9. This rule marks the following two chunks *in Indonesia boat sinking* and *in Indonesia boat capsizes* as EQUI.

A chunk can have only one alignment and once aligned, it is not considered for further alignment. However, it should be noted that our DTSim system (Banjade, Maharjan, Niraula, & Rus, 2016) supports many to many

Table 5.4: SemAligner output for a given text-pair.

Example text pairs (plain)	
S1:	Bangladesh building disaster death toll passes 500
S2:	Bangladesh building collapse: death toll climbs to 580
Example text pairs (chunked)	
S1:	[Bangladesh building disaster][death toll][passes][500]
S2:	[Bangladesh building collapse][:][death toll][climbs][to 580]
Alignment Output	
1 2 3	\Leftrightarrow 1 2 3 //EQUI //5.0 // Bangladesh building disaster \Leftrightarrow Bangladesh building collapse
4 5	\Leftrightarrow 5 6 // EQUI // 5.0 // death toll \Leftrightarrow death toll
7	\Leftrightarrow 8 9 // SIMI // 3.0 // 500 \Leftrightarrow to 580
6	\Leftrightarrow 0 // NOALI // 0 // passes \Leftrightarrow -not aligned-
0	\Leftrightarrow 4 // NOALI // 0 // -not aligned- \Leftrightarrow :
0	\Leftrightarrow 7 // NOALI // 0 // -not aligned- \Leftrightarrow climbs

alignments which we do not discuss DTSim system in this chapter. Any chunk left unpaired after applying the full set of rules is assigned the NOALIC semantic relation. Once all the chunks in the text pairs have been labeled with semantic relations, the Semantic Score scores the alignment based on the assigned semantic relation. Any chunk with NOALIC semantic relation is scored 0. The aligned chunks with EQUI, OPPO, SPE, and ALIC are invariably scored 5, 4, 4 and 0 respectively. The SIMI and REL aligned chunks may have scores between 2 and 4 depending upon the rule being applied. For example, the rule *Each chunk has a token of DATE/TIME type* assigns a score of 3 to the following alignment: *on Friday* \Leftrightarrow *on Wednesday*.

5.2.3 Alignment Output

Given an example sentence pair *S1: Bangladesh building disaster death toll passes 500* and *S2: Bangladesh building collapse: death toll climbs to 580* as input in plain text or chunked form, Table 5.4 shows the alignment output of the SemAligner tool.

The SemAligner outputs an alignment in the following format: *S1-chunk id* \Leftrightarrow *S2-chunk id* // *chunk relation type* // *chunk score* // *S1 chunk* \Leftrightarrow *S2 chunk*.

Table 5.5: F1 scores on gold and system chunked Headlines and Images training data of iSTS 2015 shared task.

Alignment	Type	Score	Alignment + Score
Headline , gold chunks			
0.884	0.639	0.787	0.613
Image , gold chunks			
0.885	0.688	0.800	0.654
Headline , system chunks			
0.821	0.546	0.715	0.523
Image , system chunks			
0.841	0.629	0.755	0.599

Unaligned chunks are identified with a 0 position index while aligned chunks are identified as a sequence of token positions in the input sentences.

5.2.4 Alignment System Evaluation

The rules of the SemAligner tool were developed using the training data of the iSTS 2015 shared task. Table 5.5 reports the F1 scores on the training data.

We evaluated the performance of the SemAligner against the gold chunked test data consisting of 378 instances of Headlines and 375 instances of Images datasets used in the iSTS 2015 shared task. The system chunks were created using our CRF chunker described in Section 5.2.1. The results are presented in Table 5.6.

Our tool performs very well for both gold and system chunks. Our system performs better or competitively in all metric categories versus the best F1 scores (Melamed, 1998) obtained for each metric category among participating systems in the shared task. The SemAligner tool provides the best performance scores (highlighted) across all performance metrics (A, T, S, T+S) in the Headlines dataset with system chunks. For gold chunks in the Headlines dataset, our system performance scores are competitive to the best performance scores across all metrics. Also, the performance scores in the Image dataset (both gold and system chunks) are comparable to the best performance scores of the participating systems in the iSTS 2015 task. Interestingly, the performance of our tool using its own

Table 5.6: F1 scores on gold and system chunked Images and Headlines test data. *MaxScore* is the best score for each metric given by any of the participating systems in the iSTS 2015 shared task.

System	Alignment	Type	Score	Alignment + Score
Headline , gold chunks				
Baseline	0.844	0.555	0.755	0.555
SemAligner	0.897	0.666	0.815	0.642
MaxScore	0.898	0.666	0.826	0.642
Image , gold chunks				
Baseline	0.838	0.432	0.721	0.432
SemAligner	0.883	0.603	0.783	0.575
MaxScore	0.887	0.614	0.796	0.596
Headline , system chunks				
Baseline	0.670	0.457	0.606	0.4571
SemAligner	0.826	0.564	0.736	0.543
MaxScore	0.782	0.515	0.702	0.509
Image , system chunks				
Baseline	0.706	0.369	0.609	0.36
SemAligner	0.852	0.568	0.749	0.539
MaxScore	0.835	0.576	0.751	0.564

chunks (system chunks) is comparable to the results obtained on the gold chunks, showing the general usability of our tool.

Our final goal is to develop a reliable interpretable layer for automated student assessment in the intelligent tutoring systems. Such a system would be useful for improving the tutee learning experience of our DeepTutor system, an ITS for tutoring students on Conceptual Physics. For example, while assessing one of the students answers to a question asked by DeepTutor in Table 5.1, the interpretable system would align the chunks and labeled them with semantic relations as:

1 \Leftrightarrow 1 // EQUI // 5.0 // Acceleration \Leftrightarrow Acceleration
2 \Leftrightarrow 2 // EQUI // 5.0 // gives \Leftrightarrow provides
3 \Leftrightarrow 3 4 5 // SPE2 // 3.0 // magnitude \Leftrightarrow magnitude and direction

The above alignment result can be used to infer that the student answer is incomplete and missing the concept *direction*. As such, the DeepTutor may

subsequently provide feedback to the student such as “*Right. In addition to magnitude, what other information is provided by acceleration?*”

However, there is still further room for improvements in our SemAligner tool, even though the tool is a top-performing system against iSTS 2015 dataset. For example, the tool has outstanding accuracy in aligning chunks (above 0.88 F-score in gold chunks and above 0.82 F-score for system chunks); but the system F-score for semantic relations labeling at best is 0.66 F-score for Headlines gold chunks.

Another thing to consider is that the tool is assessed on Headlines and Images datasets only. It does not use any student answers data for developing the iSTS model. Agirre, Gonzalez-Agirre, et al. (2016) recently released 344 annotated examples from the student answers dataset in iSTS 2016 task. We plan to utilize this data in our future versions of SemAligner tool.

5.3 Conclusion

We introduced a competitive and freely available chunk alignment tool, i.e., SemAligner that can identify semantic relations between the aligned chunks as well as compute semantic similarity scores between the chunks. It is freely available for research purposes at the SEMILAR - The Semantic Similarity Toolkits website⁴. The tool can be very useful for building an explanatory (or interpretable) layer for many NLP applications.

The SemAligner is customizable and extendible through a number of options that allow the user to configure the behavior of the tool. This Java-based tool can be used as a standalone application or as a library.

As discussed, the SemAligner provides better or comparable performance for both gold and system generated chunked text-pairs. However, there is room for improvement in the tool. Improving the ability of the system to assign better/classify semantic relation types is our next work for the future. We also plan

⁴<http://www.semanticsimilarity.org/>

to utilize the recently released annotated examples for Student answers data. The improved tool will relax the current 1:1 alignment restriction, remove ALIC relation and allow multiple alignments between the chunks.

Chapter 6

A Concept Map based Assessment of Free Student Answers in Tutorial Dialogues

In Chapter 4, we discussed how typical Semantic Textual Similarity (STS) approaches don't have interpretable functionality to explain the degree of similarity between two texts. We also presented an interpretable approach which aligns the chunks in a given pair of texts and labels them with semantic relation types and scores as proposed in the pilot interpretable Semantic Textual Similarity (iSTS) task (Agirre et al., 2015). The set of semantic relation types used were: EQUI (chunks are semantically equivalent), OPPO (chunks are opposite in meaning), SPE1/SPE2 (the chunk in the first/second sentence is more specific than the chunk in the second/first sentence), SIMI (chunks are similar but not EQUI, OPPO or SPE), REL (chunks are related but not EQUI, OPPO, SPE or SIMI), ALIC (a chunk is not aligned to any other chunk due to 1:1 alignment restriction) and NOALIC (the chunk is unrelated and has no alignment). However, the iSTS approach is just an added layer on the top of a standard STS method for interpreting the similarity score.

Moreover, neither an iSTS nor a standard STS system typically considers contextual information when computing a similarity score. In dialogue-based ITSs, it has been shown, based on an analysis of conversational tutorial logs, that contextual information is important to assess student responses (Niraula et al., 2014). They reported that 68% of pronouns in student responses were referring to entities in the previous dialogue turn or the problem description. For instance, the student answer to the tutor question in the Table 6.1 might be elliptical: "*magnitude and direction*" or containing a pronoun referring to entities mentioned earlier in the previous dialogue turn: "*it gives magnitude and direction.*" A typical

Table 6.1: A sample question and answer between student and DeepTutor with the ideal expected answer.

Description
Question: Because it is a vector, acceleration provides what two types of information?
Student Answer: Acceleration gives magnitude
Expected Answer: Acceleration provides magnitude and direction.

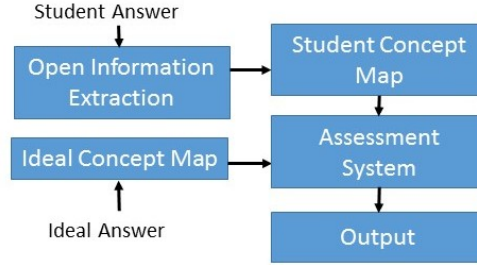


Fig. 6.1: A concept map based student answer assessment approach.

STS system might fail to deem such student answers as correct. Using an iSTS system, we might determine that both "*acceleration*" and "*provides*" in the ideal answer (expectation) are missing from/unaligned with the first student answer while "*acceleration*" is missing from/unaligned with the second student response. However, both these answers are semantically equivalent if taking context into account, i.e., the tutor question.

To address these issues, we propose a novel concept map-based approach to both better assess and interpret student free-response answers as shown in Figure 6.1. A concept map is a graphical representation of knowledge where concepts are labeled nodes and relationships between the concepts are the directed labeled edges of the graph. A concept map can be associative with no hierarchy, i.e., the concept map is a semantic network of concepts and their interrelations (Deese, 1966). Since the concept map derived from student free responses in the domain of Newtonian Physics, where our experimental data comes from, is typically

```

<Task ...>
  <ProblemDescription>A rocket is pushing a meteor with constant force. At one moment
  the rocket runs out of fuel and stops pushing the meteor. Assume that the meteor is
  far enough away from the sun and the planets to neglect gravity. How will the meteor
  move after the rocket stops pushing?
</ProblemDescription>
  <ExpectedAnswers>
    <ExpectedAnswer>
      <Id>1</Id>
      <Text>The first law is relevant because there is a zero net force acting on
      the meteor.</Text>
    </ExpectedAnswer>
    <ExpectedAnswer>
      <Id>1</Id>
      <Text>Newton first law is relevant because the problem involves the meteor on
      which no forces are acting.</Text>
    </ExpectedAnswer>
    <ExpectedAnswer>
      <Id>1</Id>
      <Text>Newton first law is relevant because the problem involves an object on
      which no forces are acting.</Text>
    </ExpectedAnswer>
    ...
  </ExpectedAnswers>
  <MisconceptionList>
    <Misconception>
      <Id>1</Id>
      <Text>meteor decreases in velocity</Text>
    </Misconception>
  </MisconceptionList>
</Task>

```

Fig. 6.2: A snippet of an XML representation of an instructional task in DeepTutor.

associative, we consider associative concept maps in our work. However, it should be noted that a concept map can also be hierarchical (Novak & Musonda, 1991) where the most general concepts are at the top.

In our approach to assessing students' natural language responses, we build an ideal concept map to represent an instructional task, say a Physics problem, based on the set of ideal steps (expectations) in the solution provided by domain experts. Figure 6.2 illustrates a snippet of an XML representation of an instructional task related with Newton's first law of motion in DeepTutor. The *ProblemDescription* describes the current problem. The *ExpectedAnswers* lists the expert-provided answers (ideal answers) for the task. It should be noted that the *ExpectedAnswer* elements having identical *Id* values are paraphrases of each other.

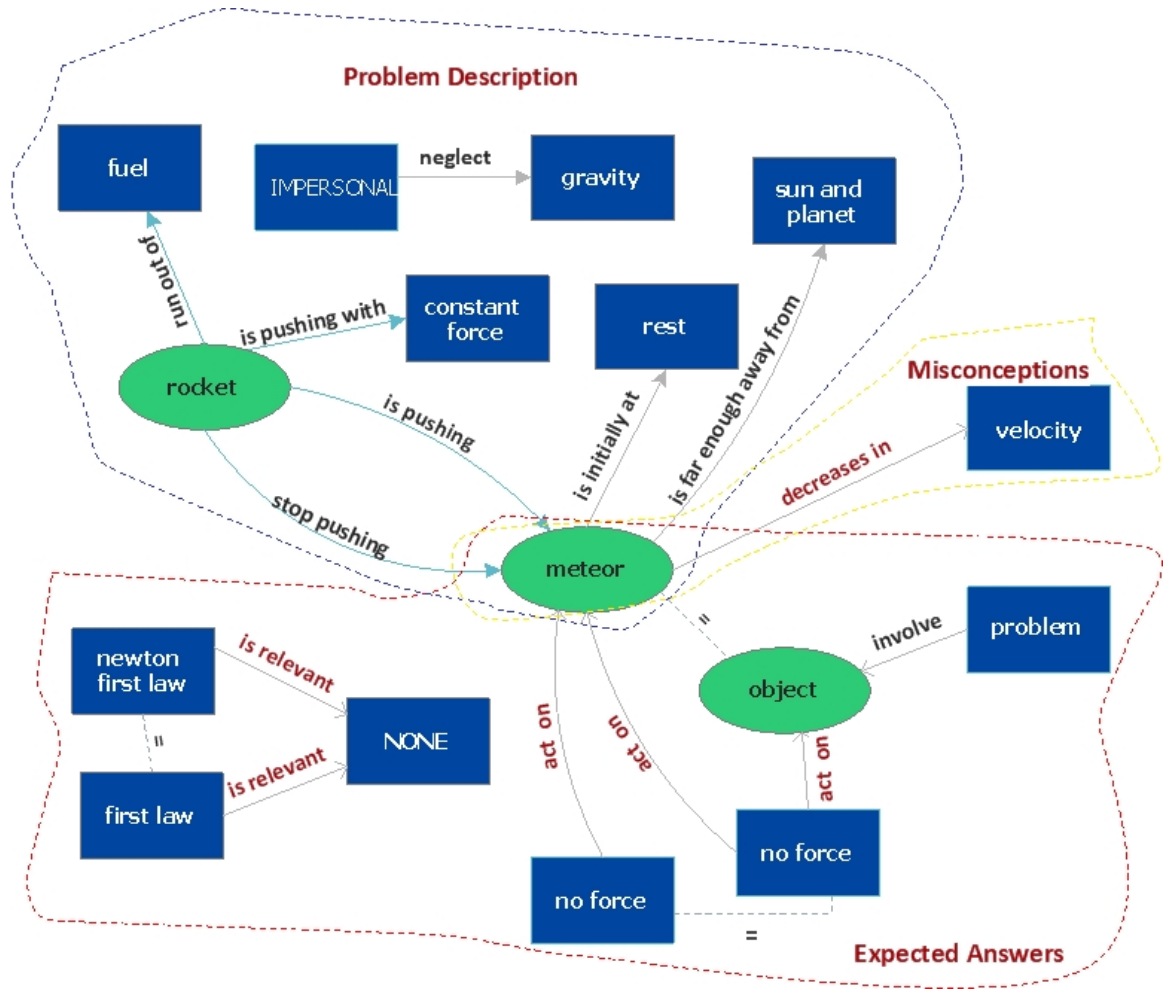


Fig. 6.3: An ideal concept map representation for the task shown in Figure 6.2.

The *MisconceptionList* lists possible misconceptions a student might have in relation to the task.

We build an ideal concept map for the problem by extracting the concepts and relationships between them from the text in the *ProblemDescription*, *ExpectedAnswers*, and *MisconceptionList* elements. The resulting concept map is shown in Figure 6.3. Similarly, we create a concept map for a student based on their responses to the task related questions. The concept map thus created can be considered a representation of the student's mental model of his understanding of the current task and target domain. The student answers may be either right or wrong which implies that parts of their concept map or graph may be correct and

<pre> <ExpectedAnswer> <Id>2</Id> <Description>general statement of Newton's first law</Description> <Pump>Can you articulate the definition or principle that helps us determine the forces?</Pump> <Text>When velocity is constant, the acceleration is zero; therefore the sum of the forces will equal zero</Text> <TaggedText> <Tuples class="java.util.ArrayList"> <Tuple requiresContextForCreation="0 1" weight="0.5" synsetId="6"> <Text>(velocity,be,constant)</Text> </Tuple> <Tuple requiresContextForCreation="0 1" weight="0.5" synsetId="6"> <Text>(acceleration,be,zero)</Text> </Tuple> <Tuple requiresContextForCreation="0 1" weight="0.5" synsetId="7"> <Text>(sum of force,equal,zero)</Text> </Tuple> </Tuples> <Comments watch="0 1"> </ExpectedAnswer> </pre>	<pre> <PredictedTuples> <Tuples class="java.util.ArrayList"> <Tuple weight="0.0"> <Text>(the sum of the force,therefore will equal,zero)</Text> </Tuple> <Tuple weight="0.0"> <Text>(the sum of the force,will equal,zero)</Text> </Tuple> <Tuple weight="0.0"> <Text>(velocity,be,constant)</Text> </Tuple> <Tuple weight="0.0"> <Text>(the acceleration,be,zero)</Text> </Tuple> </Tuples> </PredictedTuples> </pre>
a.	b.

Fig. 6.4: A comparison of an ideal concept map (a) and computer-generated concept map (b) for the ideal answer: “*When velocity is constant, the acceleration is zero; therefore the sum of the forces will equal zero*”.

parts of it may be incorrect. For example, in Table 6.1, the concept map for the expectation would be a part of an ideal concept map for the task whereas the concept map derived for the corresponding student answer would be part of a student concept map (knowledge graph) for the task. By comparing the two, we can determine which tuples (a triplet consisting of two concepts and their relationship) are matched or unmatched.

Using concepts maps for assessment leads to an important shift in the granularity of assessment. That is, breaking down an expectation into one or more tuples (a triplet consisting of two concepts and their relationship) essentially means that the unit of analysis shifts from a full sentence, i.e., an expectation, to tuples. Ideally, a concept map with a single tuple, (*acceleration, gives, magnitude*), is extracted from the student response in Table 6.1. For assessment purpose, we consider a tuple to be either *neutral* or *learning* depending upon its pedagogical value. For example, the *Expectation* in Table 6.1 is represented by two *learning* tuples (*acceleration, provides, magnitude*) and (*acceleration, provides, direction*). We may also have a composite tuple (*acceleration, provides, magnitude and direction*) that covers both *learning* tuples. Similarly, the equivalent expectation: “*Because it is a vector, acceleration provides magnitude and direction*” can be

represented by the above two *learning* tuples and one *neutral* tuple (*acceleration, is, a vector*). The notion of *neutral* and *learning* tuple is useful when assessing correct student responses that have extra information/concepts relative to the expert provided ideal answers. In fact, a good portion of student responses may contain such extra information. Banjade, Maharjan, Niraula, Gautam, et al. (2016) found that 11% of student answers contain such additional, learning-neutral information.

The advantages of using finer grain learning components in the form of tuples are many. First, we can track students' knowledge at a finer grain level leading to more subtle differences among different knowledge states. Unlike in binary assessment, the concept map approach allows tracking how much of a given expectation is covered by student response. For example, if we assess the student answer against the ideal answer in Table 6.1, we can conclude that the student has mastered 50% of the expectation. Second, we can give proper credit to student answers based on the percentage of the *learning* tuples covered. Lastly, by comparing the two maps (student's vs. expert's), we can determine missing or incorrect tuples from student answers. This finer grain assessment enables adaptive interactive learning systems to provide better feedback and to better plan the next move, e.g., providing hints about the missing learning components.

It should be noted that we first automatically extract student concept maps from their responses and then we compare them to expert-provided concept maps derived from expert-provided ideal student answer. Therefore, in this paper, we first focus on automating and developing accurate solutions for the automated extraction of concept maps from student-generated answers. Then, we propose a novel concept map-based approach which accounts for context and jointly functions as an STS and iSTS system.

6.1 Related Work

Concept maps were first proposed by Novak and Musonda (1991) to represent knowledge of science for identifying learning specific changes in children. The concept maps were developed based on the learning psychology of Ausubel (1963) whose fundamental idea was that people learn new concepts and propositions by asking questions and getting clarification of relationships between old concepts and new concepts and between old propositions and new propositions.

Concept maps have been used for many purposes. They have been used for checking students' knowledge on a topic e.g. CMap Tools (Cañas et al., 2004) and for collaborative learning of a topic (Martinez Maldonado, Kay, Yacef, & Schwendimann, 2012). Also, they have been used as instructional tools for meaningful learning, i.e., linking new information with already known information (All, Huycke, & Fisher, 2003; Horton et al., 1993; Wallace & Mintzes, 1990; Novak, Bob Gowin, & Johansen, 1983; Schmid & Telaro, 1990). Some ITSs use them as instructional tools to scaffold the learning process (Olney et al., 2012).

Concept maps have also been used for assessment. For example, students might be asked to fill in a skeleton map (Anderson & Huang, 1989), to construct a concept map (Roth & Roychoudhury, 1993; Wu, Hwang, Milrad, Ke, & Huang, 2012), or to write an essay (Lomask, Baron, Greig, & Harrison, 1992). Recently, Wu et al. (2012) developed a method that evaluates student concept maps on-the-fly and provides real-time feedback by comparing them with the expert's concept map. Also, the COMPASS (Gouli, Gogoulou, Papanikolaou, & Grigoriadou, 2004) system provides individualized feedback based on the diagnostic assessment of the learner's concept map against an ideal concept map.

From a task perspective, our assessment approach is similar to the concept map-based assessment approach of Lomask et al. (1992), with some differences. In their work, students wrote essays on two central topics in biology and then trained

teachers derived concept maps from the essays. No hierarchical structure was assumed. Similarly, in our approach, we do not assume any hierarchy. However, we automatically extract concepts maps from student-generated responses during problem-solving tutorial interactions with a dialogue-based intelligent tutoring system. In our case, the target domain is conceptual Newtonian Physics. Finally, we assess their correctness by comparing them against the corresponding ideal concept maps.

Nielsen and colleagues (Nielsen et al., 2009) decomposed the reference answer into fine-grained facets derived from the dependency parse and assessed the student response in terms of the number of facets and semantic relationships covered by the student response. However, they didn't explicitly mark which facets correspond to which parts of the student response. Our approach is different in that first we use tuples with open relations and secondly, we match corresponding tuples in both ideal and student answers. On the other hand, there are many similarities between our representation and C-rater approach for assessing short-answers (Leacock & Chodorow, 2003). In the C-rater approach, the ideal and student answers were represented as a set of normalized tuples, also known as the canonical representation. Each ideal answer was manually represented by the SMEs with essential points in canonical form (*learning* tuples) where they also mark the key phrases/words in these tuples. Our approach varies from C-rater in that the SMEs create the ideal concept map from the whole task or problem rather than ideal answer only. Therefore, the student answer can be assessed by making use of the richer context, i.e, the tuples representing the problem description, task misconceptions, and tutor question. Moreover, the SMEs describe the ideal answers with not only essential or learning tuples but also neutral and somewhat pedagogically related tuples.

We use information extraction techniques to automatically extract concept

maps. TextRunner (Yates et al., 2007) and ReVerb (Fader, Soderland, & Etzioni, 2011) uses syntactic POS tag patterns for extracting entity-relation structures, i.e., tuples in the form of (conceptA, relation, conceptB). The CREATE system (Bhattarai & Rus, 2013) generates open-relation tuples by combining the ReVerb system approach and iterative pattern and tuple-based extraction. Similarly, the Ollie system (Mausam, Schmitz, Soderland, Bart, & Etzioni, 2012) exploits learned dependency patterns to extract the tuples. The Stanford OpenIE system (Angeli, Johnson Premkumar, & Manning, 2015) first generates shorter entailed clauses from given texts using a clause splitter model and a natural logic inference system and then applies a small set of patterns to extract the tuples. These systems are geared towards building knowledge bases through open tuple extraction with a focus on extracting factual tuples from professionally written texts. As such, these systems do not produce desirable tuples for student assessment task. To address this drawback, we propose a novel open tuple extraction method, DT-OpenIE, which is more suited for the assessment task

We also incorporate context when using concept maps for assessment. There have been recent attempts that consider contextual information for assessing two short text pairs. Bailey and Meurers (2008) resolved pronouns implicitly by distinguishing between new and given concepts based on the context, i.e., a previous question. On the other hand, Banjade, Maharjan, Niraula, Gautam, et al. (2016) accounted for context by giving less weights to words already mentioned in the context. Maharjan, Banjade, and Rus (2017) handled context by using context-based count features and word-weighted similarity scores. Recently, an LSTM model has been proposed for assessing student answers in context (Maharjan et al., 2018). We combine the approaches of both Bailey and Meurers (2008) and (Banjade, Maharjan, Niraula, Gautam, et al., 2016) for implicitly resolving pronouns and ellipsis at the tuple level.

6.2 Concept Map based Approach

Using concept maps for knowledge representation is grounded on a key assumption in most cognitive theories: “*the knowledge within a content domain is well structured and organized around central concepts*”. Glaser and Bassok (1989) defined competence in a domain as “*well-structured knowledge*”. Therefore, as students acquire expertise in a domain, their knowledge becomes increasingly interconnected and resembles the subject-matter expert’s representation of the domain (Glaser & Bassok, 1989; Royer, Cisero, & Carlo, 1993). Our approach is based on those theories and (Figure 6.1) consists of three steps which we describe next.

6.2.1 Creation of Ideal Concept Maps

Currently, in dialogue-based ITSs, the subject-matter experts (SMEs) create ideal answers in the form of a set of logical steps or expectations for each instructional tasks. Unlike student concept maps, we don’t generate ideal concept maps automatically because we want ideal concept maps to be accurate representations of instructional tasks completely covering all the concepts mentioned therein.

However, the automated process might generate a concept map with some incorrect tuples and, missing some important concepts due to several issues such as POS tagging error, co-references etc. On the other hand, generating ideal concepts manually requires a large effort. So, we take a hybrid approach where we first generate the skeletal ideal maps using a syntactic pattern-based method (Fader et al., 2011) that maps ideal solutions from sets of expectations to concepts maps and then ask the SMEs to curate those automatically generated concept maps. An annotation guidelines manual was created for this purpose which experts used to correct the concept maps. We describe the step in detail in Section 6.3.2.

6.2.2 Automated Extraction of Student Concept Maps

This step extracts concept maps from student-generated answers automatically. There are several existing open information extraction (IE) tools that could be used

including the state-of-the-art Ollie (Mausam et al., 2012) and Stanford OpenIE (Angeli et al., 2015) systems. However, these systems focus on solving the Knowledge Base Problem (KBP) and as such tuples produced by these systems are not suited for the task of student answer assessment.

The Stanford OpenIE tool ignores shorter clauses not entailed from the original text. For example, given the text: “*If the acceleration of a system is zero, the net force is zero.*”, no tuples are extracted. Another issue with the tool is that its natural logic inference system tends to over-produce tuples from the text. For example, for the text “*the frictional force cancels normal force*”, the desirable tuple output is *(frictional force, cancels, normal force)*; however, the tool also generates *(frictional force, cancels, force)*, *(force, cancels, normal force)* and *(force, cancels, force)* which are all misleading for assessment. On the other hand, the Ollie system might retrieve false tuples sometimes. For example, the system retrieves the incorrect tuple *(the desk, increase its speed as, the net force)* and misses *(net force, is anymore, not zero)* when processing the following text: “*The desk increases its speed as the net force is not zero anymore*”. Also, it fails to extract any tuple from the simpler text “*Mover’s push equals friction.*”

Because of these concerns, we developed a new extraction method which exploits the strengths of these two systems. Figure 6.4 b) shows a concept map generated by our extraction system. We assessed the quality of the tuples extracted by our method using three performance metrics, *accuracy*, *coverage*, and *pedagogy* following the approach by Olney, Cade, and Williams (2011). We found that the concept maps generated by our method were significantly better than those generated by the Stanford OpenIE or Ollie system for all of the three metrics (see Section 6.3.3 for detail). We describe our tuple extraction approach next.

Clause Segmentation Model: A clause is a text segment containing a subject and a predicate. It constitutes a meaningful unit, which ideally is a

System	P	R	F
CM03	87.99	81.01	84.36
CMPR02	90.18	78.11	83.71
CM01	84.82	78.85	81.73
MP01 (Molina & Pla, 2001)	70.85	70.51	70.68
DT-CS	81.21	74.25	77.57

Table 6.2: Results of different systems on CoNLL-2001 shared task test data.

CM03 (Carreras & Marquez, 2004), CMPR02 (Carreras et al., 2002), CM01 (Carreras & Màrquez, 2001). P = Precision, R = recall, F= F-measure.

1. more force is being applied
2. since more force is being applied
3. the speed of the desk will increase
4. the speed of the desk will increase since more force is being applied

Table 6.3: An optimal clause split generated from text: *the speed of the desk will increase since more force is being applied.*

proposition. Similar to Stanford OpenIE, we extract shorter clauses from a given text and consider them candidates for tuple extraction. However, we do not use the entailment restriction or the natural logic inference system while extracting shorter clauses because of the issues discussed above.

We developed our model using the CoNLL-2001 shared task data for clause identification (Sang & Déjean, 2001). We followed the approach of Carreras and colleagues (Carreras & Màrquez, 2001) to build our clause segmentation model. First, we developed models to detect clause start and end boundaries and then a clause identification model that classifies whether a given clause candidate is a clause or not based on a confidence score. We extract clause candidates $C_{(i,j)}$ such that $j > i, word_i \in S, word_j \in E$ from the text, where the $word_i \in S$ indicates the word at position i is tagged with a clause start label S while the $word_j \in E$ means that the word at j is tagged with a clause end label E .

We evaluated the clause candidates based on confidence scores to produce a clause split from the text. A clause split is a list of consistent clauses in which

Extraction	Input Pattern
(velocity, increase, NONE)	NP + VP e.g. <i>velocity increases.</i>
(IMPERSONAL, impress, you)	To-clause e.g. He has ability <i>to impress you.</i>
(IMPERSONAL, is, no force)	NP ₁ + VP + NP ₂ , NP ₁ ∈ EX tag e.g. <i>There is no force.</i>
(1st Law, says, COMPLEX)	Attribution relation e.g. <i>1st Law says</i> that the object moves with a constant velocity
(Push, equals, friction)	NP ₁ + VP + NP ₂ , NP ₁ ∉ EX tag e.g. <i>Push equals friction</i>

Table 6.4: A list of DT patterns for tuple extraction.

clauses are either nested or not overlapping. We produced a clause split from texts using both a greedy approach (Carreras & Màrquez, 2001) and an optimal approach (Carreras et al., 2002). In the greedy approach, given an input list of clause candidates, the clause candidate with the highest confidence score is selected first and added to a clause split output and, all other clause candidates that are inconsistent with it are removed. This step is then repeated until the input list is empty. In the optimal approach, we applied a dynamic programming approach to select the optimal clause split with the highest sum aggregate of confidence scores out of all possible clause split outputs.

We used a liblinear classifier for learning with a large number of features. We used all but *Sentence Pattern* features from Carreras’ (Carreras & Màrquez, 2001) as they were found to be not discriminating enough for our liblinear model. Besides these features, we used some additional context features in our models. We also used some post-processing rules to correct the label predicted by clause start and end classifiers.

Table 6.2 provides the performance of our clause segmentation model

(DT-CS) using the optimal approach on the CoNLL-2001 shared task test data. Our system results are comparable to the top performing systems (CM01, CMPR02, and CM03). More importantly, our system extracts clauses which are reasonably suited for the student answer assessment task. Table 6.3 shows the clause split output for an example text using the optimal approach.

DT Patterns: We pass the output of the clause segmentation model through the Ollie system to generate candidate tuples. However, there are certain sentence structures which are not captured by the Ollie system that could have pedagogical value. Therefore, we applied a set of patterns to extract tuples from such sentence forms as listed in the Table 6.4. We used the special keywords IMPERSONAL and NONE to indicate the absence of first and second arguments, respectively. We used the COMPLEX keyword to denote entities which are clauses. For example, we used $NP + VP$ pattern to extract the tuple (*velocity*, *increase*, *NONE*) from the clause *velocity increases*. We used the special keyword NONE to indicate the absence of the second argument. In the case where no concept map is extracted from the text **T**, we extract the tuple as (*T*, *NONE*, *NONE*), where we consider the whole text as a complex first argument.

6.2.3 Assessment System

The Assessment System consists of two steps, namely, i) *tuple filtering* and ii) *tuple assessment*.

Tuple Filtering: Bailey and Meurers (2008) implicitly resolved co-references by ignoring words that are already known, i.e., words already mentioned in the context. However, if a student answer repeats known words/concepts, it doesn't necessarily mean that they are repeating the same information because they may use the same concepts for making different propositions. However, if a tuple in a student concept map is repeated, we can assert that the student is simply repeating the given information with high confidence because a tuple is a higher-level

construct that itself represents a proposition showing the relationship between two or more concepts. We exploit this inherent characteristic of the tuple to filter out redundant or known tuples such that the resulting student concept map contains only tuples which are most likely to carry new information or propositions.

In our approach, we consider the tuples coming from the problem description (global context) and tutor question (local context) as externally known information. We consider *neutral* tuples in the ideal concept map to be redundant as they don't cover pedagogical aspects of the target ideal answer. Therefore, if a tuple in a student concept map matches with a tuple from any of these sources, we filter out the matching tuple. For filtering purpose, we match tuples using the M_1 and M_2 tuple matching methods described below without incorporating context.

Contradictory Tuple: We consider two tuples as contradictory if they are opposite or contrasting to each other. For example, we consider (*tension, is equal to, gravity*) and (*tension, is greater than, gravity*) as contradictory tuples. However, we don't consider (*Newton's first law, is, relevant*) and (*Newton's second law, is, relevant*) as contradictory. They are related, but in the context of answer evaluation, we consider such tuples as *disjoint* tuples. In our work, if two words are in an antonymy relation in WordNet (Miller, 1995) or they match any of the sixteen rules for antonym pairs (Mohammad, Dorr, & Hirst, 2008), we consider them contradictory. We also created a domain specific antonym lookup table to address certain contrasting concepts. For example, we consider "equal", "less" and "greater" to be contradictory to each other. Similarly, we created a domain disjoint lookup list where we consider concepts such as "first" and "second" to be disjoint. We also filter out contradictory and disjoint tuples from the student concept maps.

Tuple Assessment: We assess the filtered student concept maps against corresponding ideal concept maps. We only assess if the *learning* tuples (corresponding to learning components) in the ideal concept map are

covered/matched by tuples in the student concept map. We describe our tuple matching approach below.

Given a pair of tuples, (T_1, T_2) , where $T_1 = (e_1, r_1, e_2)$, $T_2 = (e_{1'}, r_{1'}, e_{2'})$, T_1 is a tuple from ideal concept map and T_2 is a tuple from student concept map, we consider T_1 is semantically equivalent to T_2 using following three methods.

1. Matching by element-by-element (M_1): If e_1 matches $e_{1'}$, r_1 matches $r_{1'}$ and e_2 matches $e_{2'}$. We consider e_1 matches $e_{1'}$ if the set of words formed from e_1 is equal to the set of words formed from $e_{1'}$. Similarly, we define the *matches* function for element pairs $(r_1, r_{1'})$ and $(e_2, e_{2'})$.
2. Matching by bag-of-words (M_2): If the set of words formed from T_1 is equal to the set of words formed from T_2 .
3. Matching by best similarity score (M_3): If similarity score $\text{sim}(T_1, T_2) > 0.55$, where T_2 is the tuple in the student concept map that has the maximum similarity score with T_1 . 0.55 is empirically set threshold.

Incorporating Context: The filtering step discards tuples in the student answers which are deprived of new information. However, the remaining tuples with new information might also need context to match them properly. For example, we need context to accurately assess elliptical student response: "*magnitude and direction*" or student response containing pronoun: "*it gives magnitude and direction*" with the expectation in Table 6.1. In such cases, we attempt to match the tuples by incorporating context using the following two strategies. For the M_1 and M_2 methods, we account for context by considering only new concepts in student answers that were not mentioned in the previous context (Bailey & Meurers, 2008). That is, if the student response repeats the concepts mentioned in the tutor question, we ignore those concepts. We do the same for repeated concepts in the ideal answer. Specifically, we use the set of words Q from the tutor question

as context for matching element pairs (E_1, E_2) , where

$(E_1, E_2) \in \{(e_1, e_{1'}), (r_1, r_{1'}), (e_2, e_{2'}), (T_1, T_2)\}$. We consider E_1 to be semantically equivalent to E_2 under the following two cases.

1. if the set of words formed from E_1 is a proper subset of the set of words formed from E_2 and the set of unmatched words from $E_2 - E_1$ is a subset of set Q .
2. if the set of words formed from E_2 is a proper subset of the set of words formed from E_1 and the set of unmatched words from $E_1 - E_2$ is a subset of set Q .

On the other hand, in the case of the M_3 matching method, we follow a word weighting approach based on context where we don't completely discard the words in the student tuple or ideal answer tuple that are also present in context. Instead, we give less weight to such words (Banjade, Maharjan, Niraula, Gautam, et al., 2016). Specifically, we give full weight of 1 to new concepts/words and 0.4 (empirically set) to repeated concepts. Subsequently, we compute an alignment-based similarity score between the two tuples as:

$$sim(T_1, T_2) = 2 * \frac{\sum_{(r,a) \in OA} w_r w_a sim(r, a)}{\sum_{r \in T_1} w_r + \sum_{a \in T_2} w_a} \quad (6.1)$$

, where OA is optimal alignment of words between T_1 and T_2 such that $a \in T_2$ and $r \in T_1$, using Hungarian algorithm as described in Rus and Lintean (2012). The $0 \leq w_r \leq 1$ and $0 \leq w_a \leq 1$ refer to weight of the words in T_1 and T_2 respectively.

The strictness of matching relaxes as we go from the M_1 to the M_3 method. Therefore, we preferentially match *learning* tuples in the ideal concept map against the student concept in the following order: M_1 , M_2 and M_3 .

6.3 Experiment and Results

First, we evaluated the performance of our novel automated tuple extraction system, DT-OpenIE, for automatically extracting concept maps from student

```

<Utterance>
  <Speaker>DEEPTUTOR</Speaker>
  <Text>Can you articulate the definition or principle that
    helps us determine the forces?</Text>
</Utterance>
<Utterance>
  <Speaker>STUDENT</Speaker>
  <Text>Newton's law says that an object remains at rest or at
    constant velocity unless acted on by a net force.</Text>
  <CoveredExpectation>
    <Gold>T:2_6,2_7,2 F:</Gold>
  </CoveredExpectation>
</Utterance>

```

Fig. 6.5: Annotating data for binary classification. TupleIds 2.6 and 2.7 consist of expectation id 2 concatenated with synsetIds 6 and 7 respectively.

Data	# sessions	# instances
Training	21	1296
Test	20	1296

Table 6.5: Summary of Data

responses. Then, we evaluated our concept map-based approach against both binary and multi-level classification tasks. Our evaluation data is described next.

6.3.1 Data

To evaluate the concept map approach, we used student answer data from logged interactions of 41 high school students with the DeepTutor ITS (Rus, Niraula, & Banjade, 2015). During the summer of 2014, high-school students participated in an experiment on which they were given 9 different Physics problems to solve. The experiment produced 370 tutorial interactions in total (one student performed a task twice).

6.3.2 Ideal Concept Maps

Two subject-matter-experts (SMEs) manually created ideal concept maps for all 9 tasks in the data. They were provided with a reference guide for creating the ideal concept map. For each instructional task, annotators were provided with an XML file containing a skeletal concept map that needed to be checked. The map was automatically generated by using syntactic patterns (Fader et al., 2011). The annotators updated the concept maps by deleting invalid tuples, modifying tuples and by adding missing tuples.

completely(1)	concept map is completely correct.
mostly(2)	at least half of concept map is correct.
slightly(3)	at least one extracted tuple is correct.
inaccurate(4)	none of the extractions are correct.

Table 6.6: An ordinal scale with four values for rating an extracted concept map of an ideal student answer along the metric *accuracy*.

Measure	Stanford	Ollie	DT-OpenIE
Accuracy	2.71 (1.24)	2.19 (1.35)	1.89 (1.10)
Coverage	2.63 (1.28)	2.38 (1.27)	1.69 (1.12)
Pedagogy	2.52 (1.40)	2.41 (1.44)	1.70 (1.24)

Table 6.7: Mean ratings for concept maps of ideal student answers generated by different open information extraction methods. The standard deviations are provided in bracket alongside means.

While creating concept maps, the annotators also annotated the tuples with a rich set of attributes. For example, we set *symmetric* attribute to T if the first and second arguments of a tuple are interchangeable without changing its meaning, e.g., *(only force, is, gravity)* is a symmetric tuple. Similarly, we assigned tuples covering identical concepts with the same id (*synsetId*). As described earlier, we also differentiated between *learning* and *neutral* tuples depending on their pedagogical importance. We assign a weight 0 to *neutral* tuples. In case of *learning* tuples, if the expected answer has n unique *learning* tuples, we assign each of these tuples weight of $1/n$ (we consider only tuples with different ids as unique). These annotations are useful to handle variations in student answers as described in Section 6.2.3. Figure 6.4 a) shows a final human generated ideal concept map for one expectation. The annotators also set the *watch* attribute in their comments to flag their tuples for review later. After creating three concept maps, the SMEs met and compared their maps to resolve any discrepancies. A refined annotation guide was created which was then followed for the whole data annotation.

6.3.3 Automated Tuple Quality Evaluation

In order to assess the quality of the extracted tuples by different automated methods, we automatically extracted concept maps for 133 ideal student answers from the nine tasks using Stanford OpenIE, Ollie and DT-OpenIE extraction method. Then, we asked the two annotators (SMEs) to rate the generated concept maps against the ideal/gold standard concept maps. Figure 6.4 shows a comparison of an automatically generated concept map for an ideal student answer against its gold standard concept map.

The annotators rated the automatically generated concept maps along three dimensions, i) *accuracy*, ii) *coverage* and iii) *pedagogy* following the approach adopted by Olney and colleagues (Olney et al., 2011). In other words, the annotators rated the degree of correctness, completeness, and pedagogical value of the extracted tuples in the concept maps while comparing against the gold standard concept maps. We provided the annotators with an annotation guideline for the annotation. The annotators met after annotating two tasks for comparing each other annotations to resolve discrepancies if any. The guidelines were updated accordingly and used for annotating the whole data.

We used an ordinal scale of 4 values for the ratings. The Table 6.6 describes our ordinal scale for the *accuracy* measure. We used Cronbach's α to measure inter-rater reliability because of the ordinal ratings. The Cronbach's α were 0.991, 0.993 and 0.997 for accuracy, coverage, and pedagogy, respectively, which indicated a highly significant inter-annotator agreement. Another inter-annotator agreement measure, Pearson correlation, also yielded higher coefficient values of above 0.9 at significance level < 0.001 .

Results and Discussions: Table 6.7 shows the mean and standard deviations of the ratings for each of the quality measures. The mean ratings for the concept maps generated by Stanford OpenIE were 2.71, 2.63 and 2.52 for accuracy,

coverage, and pedagogy, respectively. The Ollie system-generated concept maps were slightly better with relatively lower mean ratings of 2.19, 2.38 and 2.41 for accuracy, coverage, and pedagogy, respectively. Our DT-OpenIE concept maps were the best in terms of their mean quality ratings with the scores of 1.89, 1.69 and 1.70, respectively. We performed a paired t-test significance analysis between the different extraction methods for each of the quality measures. We found that our DT-OpenIE ratings were significantly better. Compared against the Stanford OpenIE, the significances were all $p < 0.001$ for accuracy, coverage, and pedagogy, respectively. Similarly, the significances were $p = 0.004$, $p < 0.001$, and $p < 0.001$ when comparing mean accuracy, coverage, and pedagogical values of DT-OpenIE extractions against Ollies’.

We also found that accuracy, coverage, and pedagogy are significantly correlated with each other (0.85 for accuracy vs coverage, 0.8 for accuracy vs pedagogy, and 0.92 for coverage vs pedagogy).

The results are promising - the concept maps generated by our method, on an average, fall between complete and mostly accurate for all of the three quality scales. One case where the system fails to generate tuples is a list-type student response such as: "*Force of gravity and normal force*". For concept map based assessment, we extracted tuples from such text as (*force of gravity and normal force*, *NONE*, *NONE*).

6.3.4 Binary Classification Task

First, we experimented with a binary classification task in which we assess whether a particular student answer is correct or incorrect. The 41 student session data were randomly divided into training and test sets as summarized in Table 6.5.

Figure 6.5 shows an annotation example for the binary classification task. The annotators judged which of the targeted *learning* tuples are covered/uncovered by the student answer. If a tuple is covered, its tupleId is recorded in the T group.

System	F-score	Accuracy
Training Data		
WA-Sim	0.739	73.8(0.43)
WA-Sim-C	0.756	77.2(0.45)
Concept Map	0.783	78(0.53)
Test Data		
WA-Sim	0.742	74.6(0.39)
WA-Sim-C	0.752	77.2(0.40)
Concept Map	0.802	79.9(0.55)

Table 6.8: Results of different methods for binary classification. WA-Sim and WA-Sim-C are optimal word alignment-based STS system without context (Rus & Lintean, 2012) and with context (Banjade, Maharjan, Niraula, Gautam, et al., 2016) respectively. The value inside the bracket alongside the accuracy is Cohen’s Kappa.

Data	# instances	F-score	Accuracy
Training	2408	0.769	76.8(0.53)
Test	2360	0.778	77.7(0.54)

Table 6.9: Performance of concept map based approach at the tuple level.

On the other hand, if it is uncovered, its tupleId is recorded in the F group. Similarly, judges annotated full expectations as covered or uncovered by recording its expectationId either under T or F group, respectively. An expectation is covered if its all *learning* tuples are covered. In Figure 6.5, the student answer covers both learning tuples 2.6:(*velocity,be,constant*) and 2.7:(*sum of force,equal,zero*) and therefore the student answer covers the whole expectation shown in Figure 6.4.

We ran our assessment method (Section 6.2.3) against both training and test data. For comparison, we also derived results by applying the word alignment-based sentence similarity method with context (Banjade, Maharjan, Niraula, Gautam, et al., 2016) and without context (Rus & Lintean, 2012). Specifically, in the case of the context agnostic method, we first computed the similarity score by giving full weight of 1 to each word in the student and reference answer using Equation 6.1. For the context-based approach, we applied the same equation such that the words

System	F-score	Accuracy
Logistic Model	-	49.3(0.22)
GMM Model	0.58	58.2(0.40)
LSTM Model	0.62	62.2(0.45)
Concept Map	0.59	59.3(0.41)

Table 6.10: Performance on Multi-level classification task: Comparison of concept map approach against Logistic Model (Banjade, Maharjan, Niraula, Gautam, et al., 2016), GMM Model (Maharjan, Banjade, & Rus, 2017) and LSTM Model (Maharjan et al., 2018). The value inside the bracket alongside the accuracy is Cohen’s Kappa.

in the student answer or reference answer that are also present in the context, i.e., problem description and tutor question, are given lower weights.

Results and Discussions: Table 6.8 shows the performance of our concept map approach for the binary classification task. The Table 6.9 shows that our system does a better job at identifying whether a *learning* tuple is covered or not by the student concept map.

The above result implies that we can detect most of the missing *learning* tuples from the student answer. For example, we can detect that the missing tuple is *(acceleration, provides, direction)* from the student answer provided in Table 6.1. Therefore, our assessment method, if incorporated in adaptive learning systems, can enable such systems to provide more targeted relevant feedback to students and also provide useful information for planning system’s next move targeting the missing learning components. This targeted, adaptive feedback based on individual student performance could significantly improve the effectiveness of adaptive learning systems. Moreover, the results suggest that our system might work well for multi-level classification tasks as well.

6.3.5 Multi-level Classification Task

In this task, we classify the student answers into one of four correctness classes: *correct*, *correct-but-incomplete*, *contradictory* or *incorrect*. We used the DT-Grade data annotated with these four correctness classes for evaluating our approach. The

DT-Grade (Banjade, Maharjan, Niraula, Gautam, et al., 2016) dataset consists of 900 student responses sampled from logged data recorded during the same experiment with the DeepTutor system as described above.

We adapted our approach for multi-level classification task by using a simple classification rule as follows. If the student concept map covers all *learning* tuples, we classify the student answer as *correct*. If at least one of the *learning* tuples is covered, then student answer is classified as *correct-but-incomplete*. If no *learning* tuples are covered, and a *contradictory* tuple is present in the student concept map, we classify the instance as *contradictory*. If none of the above conditions satisfy, we classify the student answer as *incorrect*.

Results and Discussions: Table 6.10 shows the performance of our approach for the multi-level classification task. We note that our approach performed slightly lower than the state-of-the-art LSTM model (Maharjan et al., 2018) but performs better than other models such as logistic model (Banjade, Maharjan, Niraula, Gautam, et al., 2016) and the GMM based method (Maharjan, Banjade, & Rus, 2017).

It is evident that if we further improve the quality of extracted student concept maps, our system would perform better at tuple level which in turn would lead to better performance at both binary and multi-level classification tasks. We note that improper tuples might be extracted in some situations due to POS-tagging errors. For example, *increases* was tagged as a noun in the text “*The net forces increases and can no longer be zero*” which resulted in improper tuple extraction (*the net force increase and, can no longer be, zero*). Another issue was that the desirable student concept map was not extracted from complex elliptical student response such as “*The force of the man pushing the box and the force of friction acting on the box*” to the tutor question: “*What forces balance each other?*”. Though we handle co-reference implicitly at present incorporating explicit

co-reference resolution during tuple extraction step might further improve the performance.

6.4 Conclusion

We presented a novel automated concept map extraction method and system, called DT-OpenIE. The experiments indicate that the generated tuples are significantly better in quality than those extracted by the state-of-the-art open information extraction tools such as Stanford OpenIE and Ollie systems.

We also presented in this paper a novel concept map-based approach to assess student answers in tutorial dialogues. The approach takes context into account and implicitly handles linguistic phenomena such as ellipsis and pronouns for assessing student concept map against the ideal concept map. We combined the approaches of both Bailey and Meurers (2008) and Banjade, Maharjan, Niraula, Gautam, et al. (2016) for implicitly resolving pronouns and ellipsis at the tuple level.

We use tuples as the unit of learning to track students' knowledge at a finer grain level which enables us to better assess student answers rather than just classifying them as correct or incorrect. Moreover, our approach can easily detect missing learning components in student answers which can be used for dynamic and automatic generation of personalized of next moves such as hints. As such, adaptive tutoring systems can provide targeted adaptive feedback and scaffolding in the form of hints to the students based on their individual performance.

Our future work is to improve the concept map-based approach further. Second, we plan to exploit concept maps for dynamically providing diagnostic feedback in an automated tutoring environment and study its impact on tutoring effectiveness, i.e., on the ability of the tutoring system to induce learning gains for the learners.

Chapter 7

Conclusion and Future Work

In this dissertation, we presented our work to address research problems related to dialogue-based intelligent tutoring systems. In particular, we presented various methods and approaches to identifying effective tutorial strategies, evaluating the semantic similarity between two texts in general, and assessing student answers in tutorial dialogues. We summarize the major findings, conclusion and future work for each of the research questions below.

1. How to identify and understand effective tutorial strategies employed by successful tutors that yield tutoring sessions with learning gains?

To identify effective tutorial strategies employed by professional tutors, we described our supervised approach to mapping tutor-tutee utterances in tutorial sessions onto actions by classifying them into dialogue acts, dialogue sub-acts and dialogue modes. We presented our various analytic approaches such as profile comparison, sequence logo, discriminant sub-sequence mining, and Markov analysis to identify effective tutorial strategies. We found that the effective tutorial sessions are characterized by more *Scaffolding* and *Fading* modes on average when compared to ineffective sessions. Furthermore, the most effective sessions almost always end properly, i.e., with a *Closing* mode. On the other hand, the bottom ineffective sessions have, on average, more *ProcessNegotiation* and *ProblemIdentification*. At dialogue act level, tutors in top sessions use more expressives and prompt students more, on average, than those in the bottom sessions. We also found that any possible sequence of dialogue modes derived from a Markov process which characterizes an effective tutorial session is most likely to have more *Scaffolding* and *Fading* modes.

Further, the inferred Markov process suggests a new model for student-initiated tutorial sessions as opposed to tutor-driven sessions, which were modeled in the past.

Future Work: Our research focused on the patterns of tutorial strategies which distinguish good human tutors (or good tutoring sessions) from bad human tutors (or bad tutoring sessions). We did not study how tutoring strategies in good and bad sessions vary with different tutors (e.g. age, gender, years of tutoring experience, pay scale, etc.) as we didn't have sufficient amount of such data. It might be an interesting area of research to identify the patterns of strategies accounting for various factors of human tutors. Further, we can expand the understanding of the effective strategies by accounting for student's prior knowledge as well.

2. How to improve generic sentence similarity methods for assessing short texts?

For the semantic evaluation problem, we presented our generic semantic similarity approach to improve measuring the similarity between two short texts in English. In particular, we described our *DT_Team* system that participated in SemEval-2017 Task 1 for English track. We developed three different feature-engineered systems using SVM regression, Linear regression and Gradient Boosted regression models for predicting textual semantic similarity. Overall, the outputs of our models highly correlate (correlation up to 0.85 in STS 2017 test data and up to 0.792 on benchmark data) with human ratings. Indeed, our methods yielded highly competitive results.

Future Work: Our DT_TEAM system scores were closer to human scores (in terms of correlation) among the other top performing STS systems against the difficult English pair sentences selected by SemEval 2017 STS Task 1

organizers. However, our system also suffers from challenging issues for semantic similarity. In particular, we can further improve the Semantic Textual Similarity (STS) method by handling challenging semantic issues such as negation, semantic blending, and compositional meaning aspects.

3. How to improve automated assessment of open-ended student answers in tutorial dialogue using contextual information?

Evaluating student answers in context is particularly important in conversational tutoring environments such as DeepTutor. Therefore, we proposed two different approaches for assessing open-ended student answers in tutorial dialogue context. We first described our probabilistic Gaussian Mixture Model (GMM) model for assessing student answers. Our GMM model used multiple context-aware counts, word weighted lexical similarity scores, and word weighted alignment similarity scores as the features to assess the student answers. Our GMM model outperformed the baseline system by more than 10% in accuracy with F-score of 0.58. As such, the GMM approach is useful for capturing the variance in the student answers in the level of explicit information they contain. Next, we presented our LSTM approach to assess student answers in tutorial dialogues. We built LSTM models using pre-trained GloVe word embeddings (Pennington et al., 2014) and tri-letter encodings. Our LSTM model which used pre-trained GloVe embedding yielded the best performance on the DTGrade dataset and even outperformed the GMM method. This result suggests that our LSTM models capture the context required for assessing student answers in tutorial dialogues. Also, the pre-trained word embeddings such as GloVe which are typically developed from a large corpus of text can better represent word semantics than simple word-based or tri-letter based one-hot encodings. Another advantage of the

LSTM approach is that it does not use any domain-specific manually crafted features.

Future Work: We can explore more distinctive context-based features to improve the performance of the GMM model for assessing freely generated student answers in tutorial dialogues. Similarly, further investigation of using deep neural networks for assessing such student responses might be a promising future direction. For instance, we augmented student answer only with context in our work with the assumption that its reference answer is typically self-contained, i.e., context is not required to understand its true meaning. Therefore, a strand of future work might explore how an LSTM model performs when both student and ideal answer are augmented with context. Further, developing a joint inference model such as Markov Logic Network that incorporates different linguistic phenomena might also be a direction for future work.

4. **How to interpret the predicted similarity score and provide subsequent diagnostic feedback in tutorial dialogue contexts?**

For the interpretable similarity task, we presented our SemAligner tool based on a set of rules and lexical resources for explaining the similarity between two texts. The SemAligner identifies chunks and aligns them across the two texts indicating semantic relation and similarity score of each alignment. Our system provides better or comparable performance than the top performing system when evaluated on the iSTS 2015 shared task test data for both gold and system generated chunked text-pairs. Therefore, a rule-based system might be an effective approach for the interpretable semantic similarity task.

Future Work: Our SemAligner tool performs highly at correctly aligning the chunk pairs across the two texts. However, the performance of the system is

comparatively lower in assigning correct semantic relation labels for the aligned chunks. Therefore, we can greatly improve the performance of the system by improving the classification of semantic relation labels. Towards this end, we can explore more effective rules for classifying the semantic relation types between the aligned chunks to better explain the semantic similarity between the two texts. Another approach might be exploring a hybrid approach where we use rule-based method for correctly aligning the chunks and then, train a machine learning (ML) based classifier for predicting semantic relation labels for the aligned chunks. Further, we can combine GMM/LSTM solutions with an interpretable similarity solution, e.g., SemAligner, such that besides a holistic outcome such as Incorrect or Contradictory we also generate an explanation for the decision which would enable dynamic and automatic generation of personalized hints.

5. How to automatically extract student mental model representations from tutorial dialogues in the form of entity-relations graphs or conceptual maps and then use them for assessing student performance?

We presented our concept map approach where we automatically extracted concepts and relations from the student responses to build a concept map for each student for a given problem. The concept map is a representation of the mental model of the student understanding of the problem and target domain. Once we have the concept map for the student, we assessed the student knowledge by comparing the student concept map with the ideal concept map for the problem created by Subject Matter Experts (SMEs). Our concept map approach performed slightly better than the GMM approach but lagged behind the LSTM approach when evaluated on the DT-Grade dataset. It suggests that shifting the granularity of assessment from sentence to

finer-grained tuple level is useful for assessment. Further, we showed how the fine-grained representation of ideal answer and student answer at tuple level might be leveraged to detect missing learning components in student answers which can be used for dynamic and automatic generation of personalized hints. As such, adaptive tutoring systems can provide targeted adaptive feedback and scaffolding in the form of hints to the students based on their individual performance.

Future Work: Typically, a concept map represents things that are always true rather than true in the moment or under some conditions. However, in the case of conversational ITSs where a student interacts with the computer tutor during problem-solving, we encounter many cases where things are true only at certain instants, or under particular conditions. For example, given the statement “*If the net force is zero, the meteor moves with constant velocity*”, “*(meteor, moves with, constant velocity)*” is true only *if the net force is zero*. Similarly, given the statement “*A meteor is at rest. A rocket pushes the meteor with constant force, The rocket moves with constant acceleration*”, the tuple “*(meteor, is at, rest)*” valid at some instant t_1 is no longer true at another instant t_2 *after a rocket pushes it with constant force*, i.e., “*(meteor, moves with, constant force)*” becomes true at t_2 . Currently, we don’t model for temporal information and conditional dependence in our concept map representation. In future work, we can augment concept maps by adding such temporal and conditional dependence information. Another strand of future work might be improving the quality of the tuples extracted by the automated tuple extraction method by accounting for additional factors such as explicit co-reference resolution. Moreover, we can explore machine learning approach instead of rule-based with concept maps for student answer assessment.

References

- Abney, S. P. (1991). Parsing by chunks. In *Principle-based parsing* (pp. 257–278). Springer.
- Agirre, E., Banea, C., Cardie, C., Cer, D. M., Diab, M. T., 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., Cardie, C., Cer, D. M., Diab, M. T., Gonzalez-Agirre, 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., Banea, C., Cer, D. M., Diab, M. T., Gonzalez-Agirre, A., Mihalcea, R., ... Wiebe, J. (2016). Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 497–511).
- Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., & Guo, W. (2013). * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* sem), volume 1: Proceedings of the main conference and the shared task: Semantic textual similarity*.
- 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).
- Agirre, E., Gonzalez-Agirre, A., Lopez-Gazpio, I., Maritxalar, M., Rigau, G., & Uria, L. (2016). Semeval-2016 task 2: Interpretable semantic textual similarity. In *Semeval-2016 task 2: Interpretable semantic textual similarity*.

- semeval-2016. 10th international workshop on semantic evaluation; 2016 jun 16-17; san diego, ca. stroudsburg (pa): Acl; 2016. p. 512-24. (pp. 512–524).*
- Aleven, V., Popescu, O., & Koedinger, K. R. (2001). Towards tutorial dialog to support self-explanation: Adding natural language understanding to a cognitive tutor. In *Proceedings of artificial intelligence in education* (pp. 246–255).
- All, A. C., Huycke, L. I., & Fisher, M. J. (2003). Instructional tools for nursing education: Concept maps. *Nursing Education Perspectives*, 24(6), 311–317.
- Anderson, T. H., & Huang, S.-c. C. (1989). On using concept maps to assess the comprehension effects of reading expository text. *Center for the Study of Reading Technical Report; no. 483*.
- Angeli, G., Johnson Premkumar, M. J., & Manning, C. D. (2015). Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)* (pp. 344–354). Association for Computational Linguistics. Retrieved from <http://aclweb.org/anthology/P15-1034>
- Austin, J. L. (1975). *How to do things with words*. Oxford university press.
- Ausubel, D. P. (1963). The psychology of meaningful verbal learning.
- Bachman, L. F., Carr, N., Kamei, G., Kim, M., Pan, M. J., Salvador, C., & Sawaki, Y. (2002). A reliable approach to automatic assessment of short answer free responses. In *Proceedings of the 19th international conference on computational linguistics-volume 2* (pp. 1–4).
- Bailey, S., & Meurers, D. (2008). Diagnosing meaning errors in short answers to reading comprehension questions. In *Proceedings of the third workshop on innovative use of nlp for building educational applications* (pp. 107–115).
- 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. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 640–644).
- 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. In *Proceedings of the 11th workshop on innovative use of nlp for building educational applications* (pp. 182–187).
- 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. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 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. C., & Gautam, D. (2015). Nerosim: A system for measuring and interpreting semantic textual similarity. In *Proceedings of the 9th international workshop on semantic evaluation (semeval 2015)* (pp. 164–171).
- Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb), 1137–1155.
- Berliner, D. C. (2001). Learning about and learning from expert teachers. *International Journal of Educational Research*, 35(5), 463–482.
- Bhattarai, A., & Rus, V. (2013). Towards a structured representation of generic cconcepts and relations in large text corpora. In *Proceedings of the international conference recent advances in natural language processing ranlp*

2013 (pp. 65–73).

- Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.
- Boyer, K. E., Phillips, R., Ingram, A., Ha, E. Y., Wallis, M., Vouk, M., & Lester, J. (2011). Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden markov modeling approach. *International Journal of Artificial Intelligence in Education*, 21(1-2), 65–81.
- Brockett, C. (2007). Aligning the rte 2006 corpus. *Microsoft Research*.
- 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).
- Cade, W. L., Copeland, J. L., Person, N. K., & DMello, S. K. (2008). Dialogue modes in expert tutoring. In *International conference on intelligent tutoring systems* (pp. 470–479).
- Cañas, A. J., Hill, G., Carff, R., Suri, N., Lott, J., Gómez, G., ... Carvajal, R. (2004). Cmaptools: A knowledge modeling and sharing environment.
- Carreras, X., & Màrquez, L. (2001). Boosting trees for clause splitting. In *Proceedings of the 2001 workshop on computational natural language learning-volume 7* (p. 26).
- Carreras, X., & Marquez, L. (2004). Phrase recognition by filtering and ranking with perceptrons. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, 260, 205.
- Carreras, X., Màrquez, L., Punyakanok, V., & Roth, D. (2002). Learning and inference for clause identification. In *European conference on machine learning* (pp. 35–47).
- Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017

- task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation. In *Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)* (pp. 1–14).
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug), 2493–2537.
- Deese, J. (1966). *The structure of associations in language and thought*. Johns Hopkins University Press.
- Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–38.
- Evens, M. W., Chang, R.-C., Lee, Y. H., Shim, L. S., Woo, C. W., Zhang, Y., ... Rovick, A. A. (1997). Circsim-tutor: An intelligent tutoring system using natural language dialogue. In *Proceedings of the fifth conference on applied natural language processing: Descriptions of system demonstrations and videos* (pp. 13–14).
- Fader, A., Soderland, S., & Etzioni, O. (2011). Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 1535–1545).
- 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).
- Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). Ppdb: The paraphrase database. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 758–764).
- 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)
- Gertner, A. S., & VanLehn, K. (2000). Andes: A coached problem solving environment for physics. In *International conference on intelligent tutoring systems* (pp. 133–142).
- Glaser, R., & Bassok, M. (1989). Learning theory and the study of instruction. *Annual Review of Psychology*, 40(1), 631–666.
- Gouli, E., Gogoulou, A., Papanikolaou, K., & Grigoriadou, M. (2004). Compass: An adaptive web-based concept map assessment tool.
- Graesser, A. C., Chipman, P., Haynes, B. C., & Olney, A. M. (2005). Autotutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, 48(4), 612–618.
- Graesser, A. C., DMello, S., & Person, N. (2009). Meta-knowledge in tutoring. *Handbook of Metacognition in Education*, 361.
- Graesser, A. C., Penumatsa, P., Ventura, M., Cai, Z., & Hu, X. (2007). Using lsa in autotutor: Learning through mixed initiative dialogue in natural language. *Handbook of Latent Semantic Analysis*, 243–262.
- Graesser, A. C., Person, N. K., & Magliano, J. P. (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, 9(6), 495–522.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- Ho Lee, J., Ho Kim, M., & Joon Lee, Y. (1993). Information retrieval based on conceptual distance in is-a hierarchies. *Journal of Documentation*, 49(2), 188–207.
- Horton, P. B., McConney, A. A., Gallo, M., Woods, A. L., Senn, G. J., & Hamelin, D. (1993). An investigation of the effectiveness of concept mapping as an

- instructional tool. *Science Education*, 77(1), 95–111.
- 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).
- Jiang, J. J., & Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*.
- Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.
- 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).
- Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2), 83–97.
- Kulik, J. A., & Fletcher, J. (2016). Effectiveness of intelligent tutoring systems: a meta-analytic review. *Review of Educational Research*, 86(1), 42–78.
- 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. (1998). Combining local context and wordnet similarity for word sense identification. *WordNet: An Electronic Lexical Database*, 49(2), 265–283.
- Leacock, C., & Chodorow, M. (2003). C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37(4), 389–405.
- 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.
- Lin, D., et al. (1998). An information-theoretic definition of similarity. In *Icml*

(Vol. 98, pp. 296–304).

- Lomask, M., Baron, J., Greig, J., & Harrison, C. (1992). Connmap: Connecticut's use of concept mapping to assess the structure of students knowledge of science. In *Annual meeting of the national association of research in science teaching, cambridge, ma* (pp. 21–25).
- Maharjan, N., Banjade, R., Gautam, D., Tamang, L. J., & Rus, V. (2017). Dt_team at semeval-2017 task 1: Semantic similarity using alignments, sentence-level embeddings and gaussian mixture model output. In *Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)* (pp. 120–124).
- Maharjan, N., Banjade, R., Niraula, N. B., & Rus, V. (2016). Semaligner: A method and tool for aligning chunks with semantic relation types and semantic similarity scores. *CRF*, 82, 62–56.
- Maharjan, N., Banjade, R., & Rus, V. (2017). Automated assessment of open-ended student answers in tutorial dialogues using gaussian mixture models. In *Proceedings of the thirtieth international florida artificial intelligence research society conference (98-103)*.
- Maharjan, N., Gautam, D., & Rus, V. (2018). Assessing free student answers in tutorial dialogues using lstm models. In *International conference on artificial intelligence in education* (pp. 193–198).
- Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: System demonstrations* (pp. 55–60).
- Martin, J. H., & Jurafsky, D. (2000). Speech and language processing. *International Edition*, 710.
- Martinez Maldonado, R., Kay, J., Yacef, K., & Schwendimann, B. (2012). An interactive teachers dashboard for monitoring groups in a multi-tabletop

- learning environment. In *International conference on intelligent tutoring systems* (pp. 482–492).
- Mausam, Schmitz, M., Soderland, S., Bart, R., & Etzioni, O. (2012). Open language learning for information extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 523–534). Association for Computational Linguistics. Retrieved from <http://aclweb.org/anthology/D12-1048>
- McLachlan, G., & Peel, D. (2004). *Finite mixture models*. John Wiley & Sons.
- McNamara, D. S., Levinstein, I. B., & Boonthum, C. (2004). istart: Interactive strategy training for active reading and thinking. *Behavior Research Methods*, 36(2), 222–233.
- McNamara, D. S., Raine, R., Roscoe, R., Crossley, S. A., Jackson, G. T., Dai, J., . . . others (2012). The writing-pal: Natural language algorithms to support intelligent tutoring on writing strategies. In *Applied natural language processing: Identification, investigation and resolution* (pp. 298–311). IGI Global.
- Melamed, I. D. (1998). Manual annotation of translational equivalence: The blinker project. *arXiv preprint cmp-lg/9805005*.
- Mihalcea, R., Corley, C., Strapparava, C., et al. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Aaai* (Vol. 6, pp. 775–780).
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11), 39–41.
- Mitchell, T., Russell, T., Broomhead, P., & Aldridge, N. (2002). Towards robust computerised marking of free-text responses.

- Mohammad, S., Dorr, B., & Hirst, G. (2008). Computing word-pair antonymy. In *Proceedings of the 2008 conference on empirical methods in natural language processing* (pp. 982–991). Association for Computational Linguistics.
Retrieved from <http://aclweb.org/anthology/D08-1103>
- 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).
- Moldovan, C., Rus, V., & Graesser, A. C. (2011). Automated speech act classification for online chat. *MAICS*, 710, 23–29.
- Molina, A., & Pla, F. (2001). Clause detection using hmm. In *Proceedings of the 2001 workshop on computational natural language learning-volume 7* (p. 25).
- Morrison, D., Nye, B., Samei, B., Datla, V. V., Kelly, C., & Rus, V. (2014). Building an intelligent pal from the tutor. com session database phase 1: Data mining. In *Educational data mining 2014*.
- Nielsen, R. D., Ward, W., & Martin, J. H. (2009). Recognizing entailment in intelligent tutoring systems. *Natural Language Engineering*, 15(4), 479–501.
- Niraula, N. B., Gautam, D., Banjade, R., Maharjan, N., & Rus, V. (2015). Combining word representations for measuring word relatedness and similarity. In *The twenty-eighth international 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).

- Novak, J. D., Bob Gowin, D., & Johansen, G. T. (1983). The use of concept mapping and knowledge vee mapping with junior high school science students. *Science Education*, 67(5), 625–645.
- Novak, J. D., & Musonda, D. (1991). A twelve-year longitudinal study of science concept learning. *American Educational Research Journal*, 28(1), 117–153.
- Och, F. J., & Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational linguistics*, 30(4), 417–449.
- Ohlsson, S., Di Eugenio, B., Chow, B., Fossati, D., Lu, X., & Kershaw, T. C. (2007). Beyond the code-and-count analysis of tutoring dialogues. *Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, 158, 349.
- Olney, A. M., Cade, W. L., & Williams, C. (2011). Generating concept map exercises from textbooks. In *Proceedings of the 6th workshop on innovative use of nlp for building educational applications* (pp. 111–119).
- Olney, A. M., D’Mello, S. K., Person, N. K., Cade, W. L., Hays, P., Williams, C., ... Graesser, A. C. (2012). Guru: A computer tutor that models expert human tutors. In *International conference on intelligent tutoring systems* (pp. 256–261).
- 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., Rastogi, P., Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2015). Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification.
- 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. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).
- Pérez, D., Gliozzo, A. M., Strapparava, C., Alfonseca, E., Rodriguez, 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).
- Potthast, M., Hagen, M., Gollub, T., Tippmann, M., Kiesel, J., Rosso, P., ... Stein, B. (2012). Overview of the 4th international competition on plagiarism detection. In *Clef 2012 evaluation labs and workshop working notes papers*.
- Pulman, S. G., & Sukkarieh, J. Z. (2005). Automatic short answer marking. In *Proceedings of the second workshop on building educational applications using nlp* (pp. 9–16).
- Ritter, S., Anderson, J. R., Koedinger, K. R., & Corbett, A. (2007). Cognitive tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, 14(2), 249–255.
- Ritter, S., Fancsali, S., Yudelson, M., Rus, V., & Berman, S. (2016). Toward intelligent instructional handoffs between humans and machines. In *Workshop on machine learning for education, the thirtieth conference on neural information processing systems (nips)*.
- Rogoff, B. E., & Lave, J. E. (1984). *Everyday cognition: Its development in social context*. Harvard University Press.
- Roth, W.-M., & Roychoudhury, A. (1993). The concept map as a tool for the collaborative construction of knowledge: A microanalysis of high school physics students. *Journal of Research in Science Teaching*, 30(5), 503–534.
- Rowe, J., Mott, B., McQuiggan, S., Robison, J., Lee, S., & Lester, J. (2009). Crystal island: A narrative-centered learning environment for eighth grade

- microbiology. In *Workshop on intelligent educational games at the 14th international conference on artificial intelligence in education, brighton, uk* (pp. 11–20).
- Royer, J. M., Cisero, C. A., & Carlo, M. S. (1993). Techniques and procedures for assessing cognitive skills. *Review of Educational Research*, 63(2), 201–243.
- Rus, V., Banjade, R., Maharjan, N., Morrison, D., Ritter, S., & Yudelso, M. (2016). Preliminary results on dialogue act classification in chat-based online tutorial dialogues. In *Proceedings of the 9th international conference on educational data mining* (pp. 630–631).
- Rus, V., Banjade, R., Niraula, N., Gire, E., & Franceschetti, D. (2017). A study on two hint-level policies in conversational intelligent tutoring systems. In *Innovations in smart learning* (pp. 171–181). Springer.
- Rus, V., Conley, M., & Graesser, A. (2014). The dendrogram model of instruction: On instructional strategies and their implementation in deeptutor. *Design Recommendations for Intelligent Tutoring Systems*, 311.
- 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 *Proceedings of the 51st annual meeting of the association for computational linguistics: System demonstrations* (pp. 163–168).
- Rus, V., Maharjan, N., & Banjade, R. (2015). Unsupervised discovery of tutorial dialogue modes in human-to-human tutorial data. In *Proceedings of the third annual gift users symposium* (pp. 63–80).
- Rus, V., Maharjan, N., & Banjade, R. (2017). Dialogue act classification in human-to-human tutorial dialogues. In *Innovations in smart learning* (pp. 183–186). Springer.
- Rus, V., Maharjan, N., Lasang, T., Yudelson, M., Berman, S., Stephen, F., & Ritter, S. (2017). An analysis of human tutors actions in tutorial dialogues. In *Proceedings of the thirtieth international florida artificial intelligence research society conference*.
- Rus, V., Niraula, N. B., & Banjade, R. (2015). Deeptutor: An effective, online intelligent tutoring system that promotes deep learning. In *Twenty-ninth aaai conference on artificial intelligence* (pp. 4294–4295).
- Rus, V., Niraula, N. B., Maharjan, N., & Banjade, R. (2015). Automated labelling of dialogue modes in tutorial dialogues. In *The twenty-eighth international flairs conference*.
- Sang, E. F., & Déjean, H. (2001). Introduction to the conll-2001 shared task: Clause identification. *arXiv preprint cs/0107016*.
- Schmid, R. F., & Telaro, G. (1990). Concept mapping as an instructional strategy for high school biology. *The Journal of Educational Research*, 84(2), 78–85.
- Searle, J. (1969). *Speech acts*. Cambridge: Cambridge University Press.

- 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).
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1631–1642).
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929–1958.
- Stefănescu, D., Banjade, R., & Rus, V. (2014). Latent semantic analysis models on wikipedia and tasa. In *Language resources evaluation conference (lrec)*.
- Ștefănescu, D., Banjade, R., & Rus, V. (2014). A sentence similarity method based on chunking and information content. In *International conference on intelligent text processing and computational linguistics* (pp. 442–453).
- Steinberger, J., & Jezek, K. (2004). Using latent semantic analysis in text summarization and summary evaluation. In *Proc. isim04* (pp. 93–100).
- Sultan, M. A., Bethard, S., & Sumner, T. (2014). Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Linguistics*, 2, 219–230.
- 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 (semeval 2015)* (pp. 148–153).
- Tai, K. S., Socher, R., & Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

- Tapeh, A. G., & Rahgozar, M. (2008). A knowledge-based question answering system for b2c ecommerce. *Knowledge-Based Systems*, 21(8), 946–950.
- Tian, J., Zhou, Z., Lan, M., & Wu, Y. (2017). Ecnu at semeval-2017 task 1: Leverage kernel-based traditional nlp features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. In *Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)* (pp. 191–197).
- VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3), 227–265.
- VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4), 197–221.
- VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A. M., & Rosé, C. P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science*, 31(1), 3–62.
- Wallace, J. D., & Mintzes, J. J. (1990). The concept map as a research tool: Exploring conceptual change in biology. *Journal of Research in Science Teaching*, 27(10), 1033–1052.
- Wu, P. H., Hwang, G.-J., Milrad, M., Ke, H.-R., & Huang, Y.-M. (2012). An innovative concept map approach for improving students' learning performance with an instant feedback mechanism. *British Journal of Educational Technology*, 43(2), 217–232.
- Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., & Soderland, S. (2007). Textrunner: open information extraction on the web. In *Proceedings of human language technologies: The annual conference of the north american chapter of the association for computational linguistics: Demonstrations* (pp. 25–26).

Zhao, H., Lu, Z., & Poupart, P. (2015). Self-adaptive hierarchical sentence model.

In *Twenty-fourth international joint conference on artificial intelligence*.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., &

Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the iee international conference on computer vision* (pp. 19–27).