FEATURE COMBINATION FOR MEASURING SENTENCE SIMILARITY

Ehsan Shareghi Nojehdeh

A thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

April 2013

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Ehsan Shareghi Nojehdeh**
Entitled: **Feature Combination for Measuring Sentence Similarity**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair
Dr. Peter C. Rigby

_____ Examiner
Dr. Leila Kosseim

_____ Examiner
Dr. Adam Krzyzak

_____ Supervisor
Dr. Sabine Bergler

Approved _____
Chair of Department or Graduate Program Director

_____ 20 _____ _____
Dr. Robin A. L. Drew, Dean
Faculty of Engineering and Computer Science

Abstract

Feature Combination for Measuring Sentence Similarity

Ehsan Shareghi Nojehdeh

Sentence similarity is one of the core elements of Natural Language Processing (NLP) tasks such as Recognizing Textual Entailment, and Paraphrase Recognition. Over the years, different systems have been proposed to measure similarity between fragments of texts. In this research, we propose a new two phase supervised learning method which uses a combination of lexical features to train a model for predicting similarity between sentences. Each of these features, covers an aspect of the text on implicit or explicit level. The two phase method uses all combinations of the features in the feature space and trains separate models based on each combination. Then it creates a meta-feature space and trains a final model based on that. The thesis contrasts existing approaches that use feature selection, because it does not aim to find the best subset of the possible features. We show that this two step process significantly improves the results achieved by single-layer standard learning methodology, and achieves the level of performance that is comparable to the existing state-of-the-art methods.

Acknowledgments

I will forever be thankful to Dr. Sabine Bergler for her enduring patience and guidance. The completion of this thesis would not have been possible without her support.

I am thankful to the members of my defense committee, Drs. Leila Kosseim and Adam Krzyzak who read my thesis, and for their constructive comments and feedback.

A special feeling of gratitude to my wonderful parents and brother for their constant support and encouragement over the past two years.

I am thankful to my colleagues in CLaC lab for every single moment of laughter and frustration that we shared and for their early comments on this work. It was a priceless experience to work with you and I will never forget these days.

I would like to thank my awesome roommate, Siamak, and my friends, Mohammad Reza, Mehdi, and Nihat who were always supportive and kind.

Finally, I dedicate this work to my best friends in Iran, Elham, Morteza, and Mohadeseh who have never left my side and are very special.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

"an ancient pond / a frog jumps in / the splash of water"

Matsuo Bashō

Sentence similarity is one of the core elements of Natural Language Processing (NLP) tasks such as Recognizing Textual Entailment (RTE)[1] [Dagan *et al.*, 2006], and Paraphrase Recognition[2] [Dolan *et al.*, 2004]. Given two sentences, the task of measuring sentence similarity is defined as determining how similar the meaning of two sentences is. The higher the score, the more similar the meaning of the two sentences. An effective similarity measure should be able to determine whether the sentences are semantically equivalent or not, or how close they are, considering the variations of natural language expressions.

In addition to RTE and paraphrase recognition, an effective method to compute the similarity has many applications in a wide diversity of NLP tasks, such as Machine Translation Evaluation [Papineni *et al.*, 2002; Snover *et al.*, 2009], Text Reuse Detection [Clough *et al.*, 2002; Bendersky and Croft, 2009], Summarization [Salton *et al.*, 1997; Lin and Hovy, 2003], Question Answering [Lin and Pantel, 2001; Wang *et al.*, 2007], Information Retrieval and Extraction [Salton

[1] RTE is defined as a directional relationship between two text fragments, text (T) and hypothesis (H), and T entails H if, typically, a human reading T would infer that H is most likely true.

[2] Paraphrase Recognition is defined as identifying whether two sentences are restatement of each other with different syntax or words.

and Buckley, 1988; Baeza-Yates *et al.*, 1999], Word Sense Disambiguation [Lesk, 1986; Schütze, 1998], and Short Answer grading [Leacock and Chodorow, 2003; Pulman and Sukkarieh, 2005; Mohler and Mihalcea, 2009]. Therefore, finding an effective method for measuring the similarity between two fragments of texts is of great interest.

In 2012, with an attempt to define a framework for comparing different approaches and pipelines, the Semantic Textual Similarity (STS) shared task was proposed by [Agirre *et al.*, 2012] during the Semantic Evaluation (SemEval)-2012 workshop. The participants were asked to develop pipelines for measuring the similarity for given pairs of sentences drawn from different sources. As an example, here is a pair of sentences from the STS-2012 shared task training set:

(1.1) (a) A person is picking a komodo dragon and putting it in a box.

(b) A person is placing a baby komodo dragon into a container.

STS has two characteristics which differentiate it from the other mentioned tasks:

- **First**, instead of the binary yes/no decision in RTE or Paraphrase Recognition tasks, STS defines the notion of graded similarity within the interval of [0,5]. While 0 represents ***"on different topics"*** as in the pair:

  (1.2) (a) A man is straining pasta.

  (b) A man plays a wooden flute.

  and 5 represents ***"completely equivalent"*** as in the pair:

  (1.3) (a) The bird is bathing in the sink.

  (b) Birdie is washing itself in the water basin.

- **Second**, in entailment the relation between each H-T (Hypothesis-Text) pair is unidirectional, e.g. an apple is a fruit, but a fruit is not necessarily an apple, while in STS all the relations between pair of sentences are symmetric.

In this research, I will mainly use the dataset provided by the STS shared task-2012 organizers, in order to use their labeled data. In addition, 88 state-of-the-art systems were submitted to the task

which offers a good base for comparing the quality of the proposed system. Also, I just participated in STS shared task-2013, where 34 teams with 89 submissions were participated. Therefore, along with the 2012 datasets and its results, I will also discuss, albeit briefly, this recent shared task and the performance of my method.

1.1 Approach

Over the years different systems were proposed to measure similarity between fragments of texts. However, most of the proposed systems were based on only a single metric or resource.

Recently, more systems started to approach the similarity measurement from a new angle by combining different resources and metrics. However these new approaches, which mostly use machine learning, have their own challenges. First, finding a set of features which cover different aspect of data is a crucial step. Following Tom Mitchell's definition of machine learning[3], a feature is an individual measurable property of a experience which heuristically addresses one aspect of that experience which is relevant to the data/task we are trying to model/formulate. Second, pruning this feature space to exclude poor features, or finding the best subset of this space is a laborious task, called *Feature Selection*. In fact, feature selection is one of the fundamental steps in machine learning because sometimes the target model/function is identified by a subset of input features and not the complete feature space. In fact, poor features result in greater computational cost and may increase the chance of overfitting [Ng, 1998].

In order to answer to the first challenge, which is highly dependent on the task and the data, in this research a lexical feature space is proposed for measuring sentence similarity.

In order to address the second challenge and contrary to standard learning which builds a single model based on a single feature combination, we propose a two phase supervised learning method. It uses a combinations of lexical features and the core idea is to use all combinations of the features in the feature space during the training process, rather than just looking at one possibility for feature

[3] "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." [Mitchell, 1997]

combination. Then, a separate model is trained based on each combination and finally a second-level model will be built on top of the first-level models' predicted scores.

It is worth noting that in this research I do not apply any *feature selection* techniques such as *Exhaustive Search*, or *Forward Feature Selection* [Guyon and Elisseeff, 2003]. *Exhaustive Search* tries all different possibilities of feature subsets and selects the one that provides the best performance on training set. While the *Forward Feature Selection* begins by evaluating all feature subsets of size one, then it finds the best subsets of two features from the subsets containing f_i, the best single feature selected in the previous step, and all the remaining features, and so on.

The intuition to avoid using *feature selection* was to offer the full space of possibilities, as opposed to finding the best subset of the feature space through feature selection methods. This is beneficial because a single model, trained on one possibility of features combination, might consistently make the correct predictions for a particular class and incorrect predictions for the other classes.

The proposed approach was tested on the STS shared task-2012 and showed that it significantly improves the result that were obtained by a single-layer supervised learning approach. The proposed method performed reliably across each individual test sets (with no adjustments) and on average outperformed all 88 submitted systems. Also, I recently participated in STS-2013 shared task and ranked 4th among 34 teams and 89 submissions. This shows that the proposed method and the feature space performs reliably across different datasets of two years STS shared tasks.

1.2 Contributions

As the main contribution of the thesis, a new methodology for using features is proposed. The proposed method contrasts existing approaches in the sense that it does not aim to find the best subset of possible features. This method is beneficial specially when we are dealing with datasets that are gathered from different sources and therefore have different characteristics.

I also propose a set of lexical features for measuring sentence similarity. Each of these features covers a partial aspect of text with a possibility of overlap in the coverage. Furthermore, I propose

a schema for further analysis of the cases that the system failed to predict correctly. This schema assists us in catching linguistic phenomena which tend to occur frequently within this particular dataset. In other words, I show which aspects of language in STS are not possible to address with our lexical-oriented system.

I show that the proposed method performes reliably on the STS task and datasets (which was gathered from 5 different sources) and outperforms the single-layer standard learning methodology including the existing state-of-the-art methods. I believe that this method which lead to improvements is a very interesting methodology for further exploration on different tasks with different evaluation metrics when feasible.

1.3 Thesis Outline

The thesis is organized as follows: Chapter 2 provides background definitions that I will be using throughout this research. Furthermore, it provides a survey of related work in measuring sentence similarity in general, and in particular, covers the methodology of some of the submitted systems to STS shared task-2012. Chapter 3 presents STS training and test datasets and investigates the source of each set within these datasets. Moreover, it introduces the feature space I propose and examine in this research, and describes the learning algorithm that I selected to use in training step. Chapter 4 presents the proposed approach in details and demonstrates how this new method works and how it is generated. Chapter 5 introduces the evaluation metric which I used, describes the experiments I conducted and their results, and finally it compares the results of different experiments using two different statistical significance tests. Chapter 6 introduces the proposed error schema, and later characterizes the frequent error cases which the existing features have failed to predict. Chapter 7 presents points learned within the course of this research and proposes some future work for further research on the proposed method.

Chapter 2

Sentence Similarity

In the literature, at least two different terms are used by different authors or sometimes interchangeably by the same authors to address the same concept: semantic relatedness and semantic similarity. [Resnik, 1995] attempts to demonstrate the distinction between these two by way of an example. *cars-gasoline*, he writes, "would seem to be more closely related than, say, *cars and bicycles*, but the latter pair are certainly more similar."

It is important to note that semantic relatedness is a more general concept than similarity. Hirst [Budanitsky and Hirst, 2006] explains that, similar entities are semantically related through their similarity (*bank-trust company*), but dissimilar entities may also be semantically related by lexical relationships such as holonymy (*wheel* is-part-of *car*) and antonymy (*hot-cold*), or just by any kind of functional relationship or frequent association (*pencil-paper, penguin-Antarctica, rain-flood*). In this case the two entities are not similar, but are related by some relationship. Sometimes, this relationship may be one of the classical relationships such as holonymy, or a non-classical one as in *glass-water*, *tree-shade* and *gym-weights*. Thus, two entities are semantically related if they are semantically similar (close together in the is-a hierarchy) or share any other classical or non-classical relationships.

Within the course of this research, by "similarity" I mean "relatedness", however, in order to be

consistent with cited works and use more common terminology I use the term "similarity".

The measures that have been used to measure sentence similarity fall into two categories: syntactical and lexical. Syntactical approaches to semantic similarity detection mostly use syntactic dependency relations to construct a more comprehensive picture of the meaning of the compared texts, identifying whether a noun is considered the subject or the object of a verb. An example of a parse tree generated by Stanford Parser [De Marneffe *et al.*, 2006], representing the syntactic structure of the sentence *"The bird is bathing in the sink."*, and its dependencies are demonstrated in Figure 1 (a) and (b), respectively[1]. The following abbreviations are used in the following parse tree:

S for sentence

NP for noun phrase

VP for verb phrase

VBZ for verb, 3rd person singular present

DT for determiner

NN for noun, singular or mass

IN for preposition or subordinating conjunction

PP for prepositional phrase

And the following dependencies are used in Figure 1 (b):

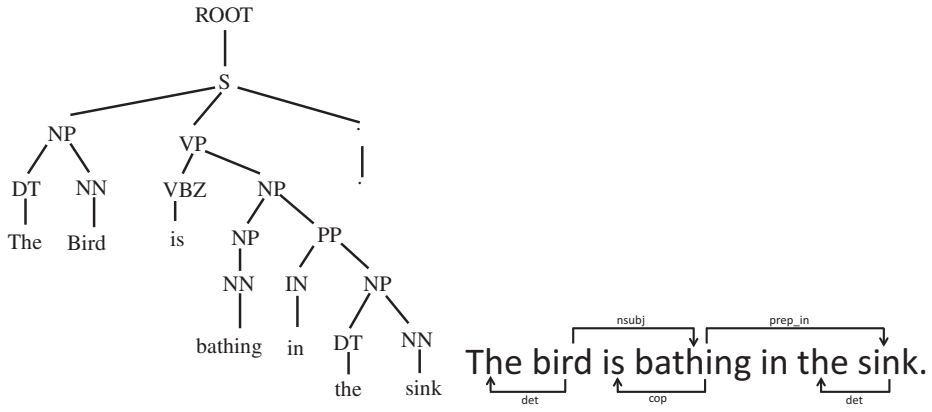
nsubj stands for the *nominal subject* relation between the verb *bathing* and the noun *bird*.

det stands for the *determiner* relation between the heads of NP ''The bird'' and NP ''the sink'' which are ''bird'', and ''sink'' respectively, and their determiner ''the''.

cop stands for the *copula* relation between the copular verb *is*, that links the subject of a clause to its complement *bathing*.

prep_in stands for the *prepositional* relation through preposition *in* between *bathing* and *sink*.

[1]Note that validating the parse tree and the dependencies generated by Stanford Parser is beyond the scope of this thesis.



(a) Stanford Parse Tree (b) Stanford Collapsed Dependencies

Figure 1: Stanford Parse Tree and Dependencies

The labeled grammatical relations generated by dependencies are extensively used in different NLP tasks. For example, Inkpen in [Inkpen *et al.*, 2006], uses *num* (relation of type number between a numeric modifier and a noun, like *num(sheep, 3)* in sentence "Sam eats 3 sheep.") and *number* (an element of compound number, like *number($, billion)* in sentence "I lost $ 3.2 billion."), and *neg* (a relation between a negation word and the word it modifies, like *neg(drive, n't)* in sentence "Bill doesn't drive.") for doing the RTE task in order to cover cases where the same word appears in different grammatical relations. Also, [Banea *et al.*, 2012] does the dependency graph alignment between two sentences for measuring the closeness of two sentences.

In addition to dependency relations, predicate argument structure (PAS) [Krestel *et al.*, 2010], which is built on top of the dependency relations, has also obtained much attention, where the verb is considered as the predicate and the subject and the object, if available, are considered as the arguments of that predicate. For example *loves(tyrannosaurus,sparrow)* is a predicate for the sentence "The tyrannosaurus loves the sparrow." [Šaric *et al.*, 2012] and [AbdelRahman and Blake, 2012] perform predicate argument alignment to catch the syntactic similarity between two sentences. These two systems will be discussed in more details in Section 2.2.

This research focuses on lexical level similarity, because they are very easy to extract and, as we will see in Section 2.2, are shown to be more effective in similarity measurement task. I will cover

different approaches for measuring lexical similarity, in detail, in the following section. Although the syntactic methods are beyond the scope of this research, some of the systems which use predicate argument structure or dependency relations will be discussed in Section 2.2.

2.1 Lexical Similarity

Two main levels for lexical features have been established: explicit level (EL), and implicit level (IL).

2.1.1 Explicit Level (EL)

Sentence similarity at the EL is based solely on the input text and measures the similarity between two sentences either by using an *n-gram model* or by reverting to *string similarity*.

1. **N-gram Models:** An n-gram is a sequence of n tokens. N-gram models are used for word prediction and also have been proposed to automatically evaluate machine translations where the task is to measure the closeness of the machine and human generated translations. The widely used n-gram models are unigrams ($n = 1$) and bigrams ($n = 2$). Another variant, the Skip-grams [Guthrie *et al.*, 2006], allows gaps of specified lengths inside n-grams. For example a 1-skip bigram model of *"I saw him"*, generates *(I/saw)*, *(saw/him)*, and *(I/him)*.

   BLEU, a widely cited and used measure proposed by [Papineni *et al.*, 2002] uses a weighted average to combine the calculated precisions for different length n-gram ($n = 1, 2, 3, 4$) matches between system translations and a set of human reference translations. A closely related method called NIST score [Doddington, 2002], was also used in machine translation evaluations sponsored by NIST over the past years, where BLEU calculates n-gram precision by considering equal weight for each n-gram unit, NIST takes into account the informativeness of a particular n-gram, which means, more weight will be put on the rarer matched n-gram. For example, using NIST, lower weight is assigned to a match based on bigram *"on the"* compared to the the correct matching of bigram *"global warming"* since the latter one is less likely to occur.

Another package which has been widely used for evaluating summaries generated by machines compared to human generated summaries is ROUGE [Lin, 2004], which offers different n-gram models such as *ROUGE-1*(unigram), *ROUGE-2*(bigram), and *ROUGE-S*(skip bigram).

In addition to machine translations and summaries evaluation, n-grams have also been used in detecting similar short passages in large documents [Lyon *et al.*, 2001] and identifying the resemblance and containment of documents [Broder, 1997], where given two documents the notions of *roughly the same* and *roughly contained* can be captured. As mentioned earlier in Chapter 1, these two tasks can be subsumed by the notion of textual similarity.

2. **String Similarity:** String Similarity considers input as a sequence of characters. One of the simplest string similarity measures is Levenshtein's [Levenshtein, 1966], which defines the distance between two strings as the minimum number of edits needed to transform one string into the other one by allowing insertion, deletion, and/or substitution of characters. In order to measure the similarity between larger lexical units e.g. phrases or code snippets, new approaches were also investigated by researchers in other domains like Plagiarism Detection or Text Reuse Detection.

   [Gusfield, 1997] proposed *longest common substring* which calculates the distance between two inputs by comparing the length of the longest consecutive sequence of characters. For example the longest common substring of two strings "ACBEFM" and "BEFTMAC" is "BEF" of length 3 (normalized: $\frac{3}{length\_of\_the\_shorter\_string} = 0.5$).

   [Allison and Dix, 1986] uses a very similar technique to that of Gutsfield, called *longest common subsequence*, except he drops the consecutiveness constraint in order to detect similarity in cases of insertions or deletions. It compares the length of the longest common sequence of characters, not necessarily consecutive ones, in order to detect similarities. For example, the longest common subsequence of the previous strings is "BEFM" of length 4 (normalized: $\frac{4}{length\_of\_the\_shorter\_string} = 0.66$).

   [Jaro, 1989] proposed an algorithm that identifies spelling variation between two inputs based

on the occurrence of common characters between two text segments at a certain distance. The distance between two inputs is calculated as follows:

$$d = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3}(\frac{m}{|S_1|} + \frac{m}{|S_2|} + \frac{m-t}{m}) & \text{if } x < 0 \end{cases}$$

where S_1 and S_2 are the inputs, m is the number of matching characters, t is half of the number of matching characters. Jaro considers two characters as matching characters only if they are same and not farther than from $\lfloor \frac{max(|S_1|,|S_2|)}{2} \rfloor - 1$. For example, the number of matching characters in previous examples is 4: 'B', 'E', 'F', 'M' and therefore the Jaro distance between them is $\frac{1}{3}(\frac{4}{6} + \frac{4}{7} + \frac{2}{4}) = 0.58$.

A year later [Winkler, 1990], a variant of *Jaro*, was proposed for name comparison and considers the exact match between the initial characters of the two inputs and is calculated as follows:

$$d_{jaro-winkler} = d_{jaro} + (\ell \times P \times (1 - d_{jaro}))$$

where d_{jaro} is the Jaro distance between two inputs, ℓ is the length of the common initial prefix (maximum of 4 initial characters), and P is the constant scaling factor, which in Winkler's original work was set to 0.1. For example, the number of common initial prefixes in previous example is 0 and therefore Jaro-Winkler distance of the previous example is $0.58 + (0 \times 0.1 \times (1 - 0.58)) = 0.58$. As you might have noticed, the reason that Jaro and Jaro-Winkler are equal, in this particular example, lies under the fact that there is no consecutive match between the initial characters of the input texts.

A few years later [Monge and Elkan, 1997] proposed a hybrid method *Monge-Elkan*, which tokenizes two inputs and finds word pairs with the highest string similarity score and then

sums up and normalizes the score and assigns it as the distance between the inputs. Monge-Elkan similarity is calculated as follows:

$$Sim_{Monge-Elkan}(S_1, S_2) = \frac{1}{|S_1|}\sum_{i=1}^{|S_1|} max(\sum_{j=1}^{|S_2|} 1 - edit\_distance(a_i, b_j))$$

where a_i, and b_j are inputs' tokens, $|S_1|$ is the number of tokens in input S_1 and $edit_distance(a_i, b_j)$ is the Levenshtein distance between two tokens. For example, the Monge-Elkan similarity between input texts "Lenovo inc." and "Lenovo corp." is:

$$Sim_{Monge-Elkan}(S_1, S_2) =$$
$$\frac{1}{2}(max(1 - edit\_distance(a_1, b_1), 1 - edit\_distance(a_1, b_2))$$
$$+ max(1 - edit\_distance(a_2, b_1), 1 - edit\_distance(a_2, b_2)))$$
$$= \frac{1}{2}(max(1 - edit\_distance(Lenovo, Lenovo), 1 - edit\_distance(Lenovo, corp.))$$
$$+ max(1 - edit\_distance(inc., Lenovo), 1 - edit\_distance(inc., corp.)))$$
$$= \frac{1}{2}(max(1 - 0, 1 - \frac{5}{6}) + max(1 - \frac{5}{6}, 1 - \frac{4}{4})) = \frac{1}{2}(1 + \frac{1}{6}) = 0.58$$

ROUGE-W[Lin, 2004], a weighted version of longest common subsequence, takes into account the number of the consecutive characters in each match, giving higher score for those matches that have larger number of consecutive characters in common. This metric was meant to measure the similarity between machine generated summaries and human generated summaries. For example, consider these strings: "A B C D E F G", "A B C D H I K", and "A H B K C I D". As it is marked with underlines, the three strings have four characters in common. Therfore, the original *Longest Common Subsequence* metric would assign equal scores to both pairs ("ABCDEFG","ABCDHIK"), ("ABCDEFG","AHBKCID"). However, using ROUGE-W, the scores for the first and second pair are 0.571, and 0.286, respectively. As I have explained earlier the fact that the length of the common consecutive characters in first pair is higher than the second one is reflected in the assigned score by ROUGE-W.

2.1.2 Implicit Level (IL)

Sentence similarity at the IL uses external resources to make up for the lexical gaps that go otherwise undetected at the EL. The *synonymy*: $(bag - suitcase)$ is an example of an implicit similarity. This type of implicit similarity can be detected via knowledge resources, such as WordNet and Roget's Thesaurus. For the more semantically challenging relations, for example $(sanction - Iran)$, which the aforementioned knowledge resources do not provide any relations/links, co-occurrence-based measures that can be calculated based on any corpus of text such as Wikipedia, or newspaper articles are more robust.

1. **Knowledge-based Measures:** Before going any further, I want to clarify one point. The term *knowledge* is often used to refer to broader range of resources rather than dictionaries. For example, one may use knowledge base to refer to Wikipedia, a Database of geographical-information, etc. However in this research I only consider lexical resources, WordNet and Roget's Thesaurus, as the knowledge base.

   WordNet : WordNet is a lexical database of more than 150000 English words. The base block of WordNet is synset, which is a set of synonyms representing the same concept. These synsets are grouped into nouns, verbs, adjectives and adverbs, with links within these groups and not between them. Most of the synsets are connected through a number of semantic relations such as *hyponymy* (dolphin *is-a* mammal) and its inverse, *hypernymy*, *holonymy* (car-door *is-part-of* car) and its inverse, *meronymy* for nouns, *hypernymy*, *troponymy*(*to lisp* is a troponym of *to talk*), *entailment* (*to sleep* is entailed by *to snore*), and *cause to* for verbs, *related nouns* and *similar to* for adjectives, and *root adjectives* for adverbs. Both nouns and verbs are organized into hierarchies, defined by hypernym relations. Each hierarchy in WordNet can be visualized as tree with general concepts associated with a root and more specific concepts associated with leaves. An example of WordNet entry, representing this hierarchy, for the word "Choice" is shown in below:

   ```
   Choice (Sense 1)
   ```

```
=> choice, pick, selection

 => decision making, deciding

  => higher cognitive process

   => process, cognitive process, ..., cognitive operation

    => cognition, knowledge, noesis

     => psychological feature

      => abstraction, abstract entity

       => entity
```

Based on this hierarchical structure of WordNet, six similarity measures have been proposed [Pedersen *et al.*, 2004]. However due to the constraint that, hypernymy/hyponymy relations in WordNet do not cross part of speech boundaries, these measures are limited to judgments within noun and verb pairs. The reason that most of the measures based on WordNet do not support adjective and adverbs lies under the fact that the taxonomy of adjectives and adverbs is not as rich as the taxonomy of verbs and nouns. This is because adjectives and adverbs do not have the hierarchical structure (similar to hypernymy/hyponymy) that verbs and nouns do.

Among these similarity measures, three of them are based on the *information content* of the least common subsumer concept [2] that subsumes both of the compared concepts. As an example, according to the hierarchy presented in Figure 2, the *lcs* of words *apple* and *cucumber* is *green goods*, and the *lcs* of *apple* and *pasteurized milk* is *food*.

[2] [Pedersen *et al.*, 2004] uses *least common subsumer*(lcs), [Budanitsky and Hirst, 2006] uses *most specific common subsumer* and [Wu and Palmer, 1994] uses *lowest super-ordinate*(lso) to refer to the same concept. Within the course of this research I will use the term *least common subsumer*(lcs).

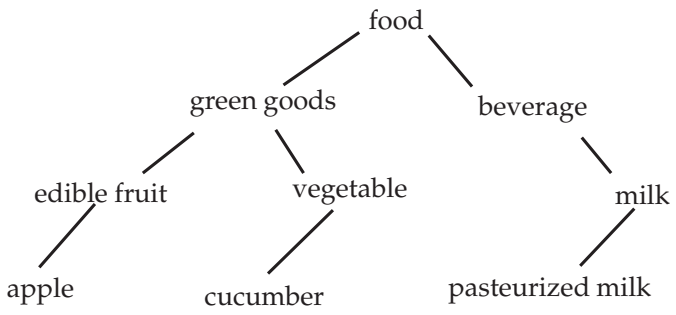


Figure 2: An example of the hypernymy/hyponymy hierarchy in WordNet.

[Resnik, 1995] calculates the information content (IC) of concepts by using frequencies gathered from Brown Corpus of American English. Where the information content of a concept c_1 is calculated as

$$IC(lcs(c_1)) = -logP(lcs(c_1))$$

where $P(lcs(c_1))$ is the probability of encountering concept c_1. Using information content, he defines the similarity of two concepts as the value of information content of their least common subsumer (lcs):

$$sim(c_1, c_2) = -\log p(lcs(c_1, c_2)) = IC(lcs(c_1, c_2))$$

[Jiang and Conrath, 1997] define the distance between two concepts to be the sum of the difference between the information content of each of the two given concepts and their least common subsumer:

$$dist(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \times IC(lcs(c_1, c_2))$$

where the similarity of two concepts is equal to $\frac{1}{dist(c_1,c_2)}$.

[Lin, 1998b] defines a measure of similarity that uses Resnik's measure but scales it using

the sum of the information content of the compared concepts:

$$sim(c_1, c_2) = \frac{2 \times IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)}$$

The other three measures (including the shortest path measure) are based on path lengths. [Wu and Palmer, 1994] find the depth of the least common subsumer, and then normalize that by the sum of the depths of the two compared concepts in order to compare what they called *conceptual similarity*:

$$sim(c_1, c_2) = \frac{2 \times depth(lcs(c_1, c_2))}{len(c_1, lcs(c_1, c_2)) + len(c_2, lcs(c_1, c_2)) + 2 \times depth(lcs(c_1, c_2))}$$

where $len(c_1, lcs(c_1, c_2))$ is the number of nodes on the path from c_1 to $lcs(c_1, c_2)$ and $depth(lcs(c_1, c_2))$ is the number of nodes on the path from $lcs(c_1, c_2)$ to root.

[Leacock and Chodorow, 1998], find the shortest path between two concepts and normalizes that by the maximum depth of the hierarchy in which these two concepts occur:

$$sim(c_1, c_2) = -\log \frac{len(c_1, c_2)}{2 \times \max depth(c)}$$

Pedersen [Pedersen *et al.*, 2004] categorizes the rest of the existing measures,[Hirst and St-Onge, 1998], [Banerjee and Pedersen, 2003], and [Patwardhan, 2003], as more general ones in that they can go beyond the part of speech boundaries, and they are not only limited to is-a relations. [Hirst and St-Onge, 1998] introduces a measure that considers many other relations. It classifies all WordNet relations as horizontal, upward, or downward. Upward relations connect more specific concepts to more general ones, while downward relations links more general concepts to more specific ones. Horizontal relations keep the same level of specificity. For example *is-a* is an upward relation while *antonymy* is a horizontal relation. Using these notions of direction they measure the relatedness between a pair

of concepts by finding a path that is not long and does not have too often changes in direction. Assigning 16 as the indicator of strong relation between two concepts, they define the weight of the relation between two concepts as follows:

$$path\_weight = 16 - path\_length - number\_of\_changes\_in\_direction$$

Incorporating WordNet glosses (brief definitions for each concept or sense in WordNet), [Banerjee and Pedersen, 2003](*lesk*) and [Patwardhan, 2003](*vector*) measures use the text of each gloss as a representation for the corresponding concept. *lesk* measures the relatedness of two concepts by calculating the number of overlaps between the glosses of the two concepts, as well as concepts that are directly linked to them in WordNet hierarchy. *vector* measure creates a cooccurrence matrix from a corpus (collection of documents) made up of the WordNet glosses. In this co-occurrence matrix, rows correspond to terms and columns represent documents and each cell contains a $tf * idf$ weight of the corresponding word in the corresponding gloss.

$tf * idf$ [Salton *et al.*, 1975] is a numerical statistic which reflects how important a word is to a document in a collection of documents. tf for term w_i in document d_j is equal to its frequency in d_j, and idf stands for inverse document frequency calculated by $idf = \log(\frac{N}{n_i})$, where N is the number of documents in the corpus and n_i is the number of documents in which the term w_i occurred. Having these tf and idf values, the $tf * idf$ weight of the word w_i in document j is calculated as $(1 + \log tf_{ij} \times \log \frac{N}{n_i})$ if the term has occurred in document j, and otherwise the $tf * idf$ weight is zero.

In the case of *vector*, each gloss represents a document and a corpus of WordNet glosses represents a corpus of documents. In this generated co-occurrence matrix each word (ignoring stop words, such as *is*, *the*, *all*) used in a WordNet gloss has an associated context vector. Each gloss is represented by a gloss vector that is the average of all the context vectors of the words found in that gloss. Relatedness between concepts is then

measured by calculating the cosine similarity between a pair of gloss vectors. Where the cosine similarity is calculated as the inner product of two vectors (after transforming them into unit vectors) and measures the cosine of the angle between them, and can vary between 1 and -1, where 1 means two vectors are pointing to the same direction and therefore the corresponding words are similar, while -1 means they point to the opposite directions and are dissimilar.

WordNet has been widely used in different NLP related tasks. [Inkpen *et al.*, 2006] uses WordNet to generate features for recognizing textual entailment by considering overlaps between synsets of words and the *antonymy* relation. More systems that made use of WordNet for measuring sentence similarity will be discussed in Section 2.2.

Roget's Thesaurus Roget's Thesaurus is another lexical resource and its electronic version was created by [Jarmasz and Szpakowicz, 2003]. It is based on well-crafted concept classification and was created by professional lexicographers. Its taxonomy has eight Classes, while the first three, *Abstract Relations*, *Space* and *Matter*, cover the external world and the rest, *Formation of ideas*, *Communication of ideas*, *Individual volition*, *Social volition*, *Emotion*, *Religion and Morality* cover the internal world of human beings.

Roget's Thesaurus has a nine-level ontology. These levels, from top to the bottom are, 9 classes, 39 sections, 79 sub-sections, 596 head groups, and 990 heads, parts of speech, paragraphs, semicolon groups, and words. The main concepts in this ontology are considered to be represented by heads. An example Roget's entry for the word "Choice" is shown in below, where the first 6 lines represent class (WORDS RELATING TO THE VOLUNTARY POWERS), section (INDIVIDUAL VOLITION), subsection (VOLITION IN GENERAL), head group (ACTS OF VOLITION), head (ChOICE), part of speech (NOUNS), and each fragment ending with a semicolon is a semicolon group, and each fragment ending with dot represents a paragraph:

```
WORDS RELATING TO THE VOLUNTARY POWERS
```

INDIVIDUAL VOLITION

VOLITION IN GENERAL

ACTS OF VOLITION

ChOICE

NOUNS

choice, option; discretion (volition); preoption;

alternative; dilemma, embarras de choix; adoption,

cooptation; novation; decision (judgment).

election, poll, ballot, vote, voice, suffrage,

plumper, cumulative vote; plebiscitum, plebiscite,

vox populi; electioneering; voting; elective franchise;

straight ticket [U.S.].

selection, excerption, gleaning, eclecticism; excerpta,

gleanings, cuttings, scissors and paste, cut and paste;

pick (best).

preference, prelation; predilection (desire).

[Jarmasz and Szpakowicz, 2003] define the following schema as the representative for semantic distance in Roget's Theasurus. According to their schema, the distance between two terms decreases as the the common head that subsumes them moves from top to bottom and becomes more specific:

distance 0 : in the same Semicolon Group. Example: *choice* - *option*

distance 2 : in the same Paragraph. Example: *choice* - *alternative*

distance 4 : under the same Part of Speech. Example: *choice - election*

distance 6 : under the same Head. Example[3]: *choice (Noun) - embrace (Verb)*

distance 8 : under the same Head Group. Example[4]: *choice - necessity*

distance 10 : under the same Sub-section. Example[5]: *choice - motive*

distance 12 : under the same Section. Example[6]: *choice - intention*

distance 14 : under the same Class. Example[7]: *choice - freedom*

distance 16 : in the Thesaurus. Example: *choice - accounts*

Roget's does not have one of the WordNet's major drawbacks, which is the lack of links between parts of speech. Therefore, Roget's Thesaurus can link a noun (e.g. bank) to a verb (e.g. invest) by finding a common Head (e.g. 784 Lending). Following this last point, I believe that Roget's-based relatedness measure can augment the similarity measures that are based on WordNet. I will examine this during my experiments.

The thesaurus was used by [Morris and Hirst, 1991] for recognizing semantic relationships between words, and by [Jarmasz and Szpakowicz, 2003] for measuring semantic similarity of words.

2. **Co-occurrence-based Measures**: Co-occurrence-based measures, offer a larger coverage than the previously mentioned lexical resources; named entities (e.g. Microsoft, Google), technical terms, slang-language, as well as other explicit relations that may exist between words are detected based on the co-occurrence of words in the same context. It is worth noting that co-occurrence-based measures also introduce noise. There are two categories of co-occurrence-based measures: *semantic analysis models* and *vector-based* models. The semantic analysis models acquire abstract concepts from a large corpus and link words to these concepts/latent-topics, ultimately going beyond raw documents, which is the draw-back of the vector-based model [Hassan and Mihalcea, 2011]. In this section, I will briefly review the $tf*idf$ *vector-based*

[3]Under the head "Choice"
[4]Under the head group "Acts of Volition"
[5]Under the subsection "VOLITION IN GENERAL"
[6]Under the section "INDIVIDUAL VOLITION"
[7]Under the class "WORDS RELATING TO THE VOLUNTARY POWERS"

model and will discuss some *semantic analysis* models such as latent semantic analysis (LSA), explicit semantic analysis (ESA), and a LSA variant, Probabilistic LSA (PLSA) in more details.

TF-IDF Model: In vector-space based models [Salton and McGill, 1986], which are widely used in Information Retrieval [Baeza-Yates *et al.*, 1999] and Natural Language Processing [Manning and Schütze, 1999; Jurafsky *et al.*, 2000], documents are represented as a binary or $tf * idf$ word vectors and the similarity between words/sentences/documents is measured by calculating similarity metrics like cosine similarity between their vectors. For example assume that we are a given the count matrix in below, where each row corresponds to a word and each column corresponds to a magazine labeled with their domain, and the numbers represent the frequency of the corresponding words in each magazine:

	General	Science	Sport
cancer	10	23	0
champion	2	0	57
gossip	24	0	6
storm	112	37	0

Table 1: Term-Document count matrix

and by transforming this count matrix into a $tf * idf$ weight matrix and normalizing the vectors using their length (so that each vector is a unit vector), we get the following matrix:

	General	Science	Sport
cancer	0.6	0.8	0
champion	0.25	0	0.96
gossip	0.89	0	0.52
storm	0.8	0.6	0

Table 2: Term-Document $tf * idf$ weight matrix

Using these normalized vectors (rows in table) one can calculate the cosine similarity of two terms (rows) in this table. For example the following calculation shows that the similarity between "cancer" and "storm" is higher than the similarity of "cancer" and

"gossip":

$$sim(cancer, storm) = 0.6 \times 0.8 + 0.8 \times 0.6 + 0 \times 0 = 0.96$$

$$sim(cancer, gossip) = 0.6 \times 0.89 + 0.8 \times 0 + 0 \times 0 = 0.53$$

Latent Semantic Analysis (LSA): LSA [Landauer *et al.*, 1998][8] generates a latent topic-model based on the term-document matrix by reducing its rank. As a result, in the new space, the similarity between the vectors of two terms will be higher if they tend to frequently co-occur in the same context (latent topics) and not necessarily in the same documents. It takes the term-document matrix, whose rows correspond to terms and whose columns represent documents. Each cell in this sparse matrix contains the $tf * idf$ weight of the corresponding term. Then, the linear algebraic method called Singular Value Decomposition (SVD) is applied to this matrix to reduce its rank while preserving the similarity structure among rows. Skipping the mathematical description, the idea of LSA is to build term-latent concept matrix (term-latent semantic space) out of the term-document matrix (term-document space). While the explanation from the linguistic point of view is not very clear, from a mathematical point of view, the SVD algorithm maps vectors from higher dimension to a lower dimension while satisfying the condition that independent vectors will remain independent in the new space. This reduction generates an abstraction of meaning by collapsing similar terms and discarding noisy and irrelevant ones. As an example, assume that we are given the following sentences:

(2.1) i. *All Romans are either loyal to Caesar or hate Caesar.*

ii. *Marcus is a man.*

iii. *All Pompeians are Romans.*

iv. *Marcus is a Pompeian.*

we first generate the normalized $tf*idf$ matrix, assuming that each sentence is a document

[8] I used Lucene for the indexing and calculating $tf * id$ and Colt library for applying singular value decomposition in my Implementation for LSA.

(ignoring stop words):

	i	*ii*	*iii*	*iv*
caesar	1	0	0	0
hate	1	0	0	0
loyal	1	0	0	0
man	0	1	0	0
marcus	0	0.7	0	0.7
pompeian	0	0	0	1
pompeians	0	0	1	0
romans	0.7	0	0.7	0

If we decompose this matrix and reduce its ranks and compose it again, we will get the following matrix:

	i	*ii*	*iii*	*iv*
caesar	0.949	0	0.2	0
hate	0.949	0	0.2	0
loyal	0.949	0	0.2	0
man	0	0.501	0	0.501
marcus	0	0.701	0	0.701
pompeian	0	0.501	0	0.501
pompeians	0.22	0	0.051	0
romans	0.819	0	0.19	0

the dimension reduction step has collapsed the component matrices in such a way that words that occurred in some contexts now appear with greater or lesser estimated weights, and some that did not appear originally now do appear, at least fractionally. For example if we look at the vector of "pompeians" and "caesar" in the original matrix and compare it with their corresponding vectors in the new space, we see that in the new space the similarity of these two terms, using cosine similarity, is not zero anymore (in the following computation, the vectors in the new space are normalized):

$$sim_{original}(pompeians, caesar) = 0 \times 1 + 0 \times 0 + 1 \times 0 + 0 \times 0 = 0$$

$$sim_{new}(pompeians, caesar) = 0.96 \times 0.98 + 0 \times 0 + 0.22 \times 0.21 + 0 \times 0 = 0.98$$

As it can be seen in the new space the two mentioned terms' corresponding vectors are

very close to each other. Since we have a very small number of sentences we can actually look at these sentences and notice the fact that "pompeians" and "caesar" do not co-occur with each other, but the word "Romans" co-occurs with both of them. The same thing can be said about the two other terms "man" and "pompeian", while they don't co-occur with each other, the term "Marcus" co-occurs with both of them. As I mentioned earlier, although SVD is mathematically robust, the way that the rank reduction changes the original weights does not have any linguistic interpretation.

Probabilistic Latent Semantic Analysis (PLSA): A year later, Hofmann [Hofmann, 1999a] proposed an approach called Probabilistic Latent Semantic Analysis (PLSA) which provides a sound statistical foundation for LSA. While both methods start from term-document co-occurance matrix and decompose it into a multiplication of three matrices, the way of building these matrices and the justification beyond it is very different.

Skipping the mathematical details, Hofmann's method is an iterative algorithm, which given the original matrix, in the first iteration by assuming uniform distribution over topics randomly picks k documents (as a representative for k topics), then it calculates the conditional probability of having each term, given each of the selected topics. This produces a term-topic matrix. Next, it uses the calculated conditional probabilities to compute the probability of having each of the selected topics in each of the original documents. This builds a topic-doc matrix. In his method he also considers weight vectors to automatically adjust the importance of each topic. In the end of the first iteration, it updates the weight vectors and finds new centroids and set them as the topics for the second iteration and so on. This procedure halts when the difference between old and new topics are less than a predefined value. In simple words, his theory is, when an author wants to write a document, he thinks about topics that he wants to cover in his article and then for each topic he thinks about words that he needs to select for writing that topic. While his similar method for indexing called Probabilistic latent semantic indexing

[Hofmann, 1999b] has been proven to be very useful in information retrieval due to its lack of clarity in terms of linguistic interpretation, little attention has been paid to PLSA in the semantic similarity researches.

Explicit Semantic Analysis (ESA): ESA [Gabrilovich and Markovitch, 2007] uses encyclopedic knowledge to generate a semantic interpretation of words. It maps the input text into a weighted sequence of Wikipedia articles ordered by their relevance to the input text. ESA uses predefined concepts represented by Wikipedia articles' titles.

First[9], it represents each Wikipedia concept as a vector of length N (size of the Wikipedia's vocabulary) whose entries are $tf * idf$ values of the corresponding term. Next, it builds an *inverted index*, which maps each individual word in Wikipedia to a list of concepts in which it appears. Then for each term, by considering its $tf * idf$ score in each of the listed concepts, it sorts the list. This procedure is called *building semantic interpreter*. Finally, given an input text, it transforms it into tf-idf vector representation and iterates over its words and for each word it iterates over the top ten concepts in the sorted list in order to generate the final weighted list of the concepts for the input. This procedure is called *using semantic interpreter*. See Figure[10] 3.

For example, Let $\vec{V} = [v_1, v_2, ..., v_N]$ be the tf-idf vector for the input text, where v_i is the tf-idf weight of word w_i (if it exists in the input text, otherwise $v_i = 0$) and N is the vocabulary size of the Wikipedia. Let $\vec{K_i} = [k_{i1}, k_{i2}, ..., k_{iM}]$ be the vector of inverted index for term w_i, where M is the number of concepts in Wikipedia (in ESA only the top ten concepts are used) and k_{ij} represents the strength of the association of word w_i with the concept c_j, which equals to the tf-idf value of the term w_i in the Wikipedia article representing concept c_j. Having these vectors, the weighted vector of the concepts for a given input text of size n is generated as $\sum_{i=1}^{n} v_i.\vec{K_i}$.

[9] I used Lucene for indexing and creating the inverted index and MySQL for Storing Wikipedia's articles in my implementation for ESA. Also, because of the resource limitation, I used Wikipedia's XML dumps from 2004.

[10] The figure was taken from the paper introduced ESA [Gabrilovich and Markovitch, 2007].

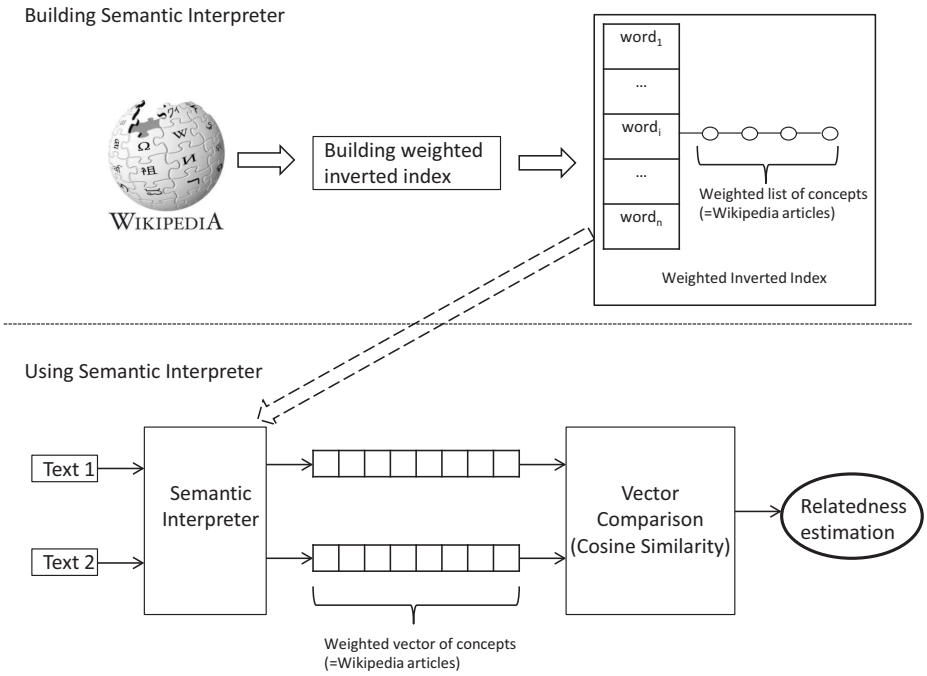


Figure 3: Explicit Semantic Analyzer

Having this weighted vector of concepts for each input, the semantic relatedness between these inputs can be calculated by applying the cosine metric. For example Table 3 demonstrates the top ten concepts that ESA returns for the pair 723 of MSRpar dataset from the STS-2012 shared task's training set:

	Schroeder cancelled his Italian holiday after Stefani refused to apologise for the slurs, which came after Berlusconi compared a German politician to a Nazi concentration camp guard.	Stefani's remarks further stoked tension after Italian Prime Minister Silvio Berlusconi last week compared a German member of the European Parliament to a Nazi concentration camp guard.
1	Patricia Schroeder	Forza Italia
2	Stefanie Powers	Silvio Berlusconi
3	USS Schroeder (DD-501)	Stefanie Powers
4	Forza Italia	Socialist Party New PSI
5	Nazi concentration camps	Lamberto Dini
6	Schroeder (Peanuts)	Martin Schulz
7	Silvio Berlusconi	Stokes County, North Carolina
8	Gerald Schroeder	Vic Schroeder
9	Vic Schroeder	Nazi concentration camps
10	Gwen Stefani	Silvio O. Conte

Table 3: Top ten concepts returned by ESA for the sample pair of sentences.

Others: Lin proposed a Dependency-based Distributional Thesaurus [Lin, 1998a], which uses distributional patterns of word dependencies in a large corpus to measure the similarity between pairs of words. In Lin's distributional similarity measure, given a word, the proposed thesaurus lists up to 200 most similar words and their similarity scores.

Another distributional measure is Pointwise Mutual Information (PMI) [Turney and others, 2001], which simply calculates the similarity between two words by the probability of seeing them together within a window of size N in a corpus divided by the probability of having them separately. Given the fact that PMI only works well in very large corpora [Inkpen, 2007], to obtain these probabilities they used *AND* query through AltaVista advanced search query and used the number of return hits.

To address the aforementioned drawback of PMI, the *second-order co-occurrences PMI (SOC-PMI)* was proposed by [Islam and Inkpen, 2006]. SOC-PMI looks at the words that co-occur with the two target words and sorts them in two lists corresponding to each individual target word. Then for words that are common in both lists it aggregates the PMI values from the opposite list to calculate the similarity of the target words.

2.2 Related Work

So far, I have introduced methods that have been used separately to measure semantic relatedness. As I demonstrated in previous sections, each of these methods address a partial aspect of text meaning. Therefore, building a hybrid system, which makes use of a combination of methods to use best of each of these methods sounds promising. In this section to the end of this chapter, I briefly discuss some of the state-of-the-art systems.

Recent work by [Bär *et al.*, 2012] (UKP) uses machine learning techniques using a variety of features produced by n-grams and string matching, ESA-based vector comparison, and WordNet similarity measures. In addition to the mentioned features, they made use of BIUTEE textual

entailment system[11] [Stern and Dagan, 2011] to use the entailment result as a feature, which wasn't successful and the feature wasn't selected for the final features set. Also, in order to populate the training set, they used Moses [Koehn *et al.*, 2007], a statistical machine translation system that allows one to automatically train translation models for any language pair, to translate each of the sentences from English to Dutch, German, Spanish and then translate it back to English. Their intuition was, by doing this transformation additional lexemes will be introduced which may make the relatedness clearer.

Similarly [Šaric *et al.*, 2012] (TakeLab) uses a support vector regressor as the learning algorithm, LSA and two of the WordNet-based measures with a combination of features covering n-gram overlaps. For the purpose of considering words' importance, they also incorporate information content of each word in calculating the cosine similarity. They also consider predicates of type *subject-verb-object*, and tried to align corresponding arguments between two sentences. Additionally, they compare dependencies argument for cases that there is a match between the dependency types. They claim that their system's performance, raised by including named-entity and number matching as separate features in the feature space.

[Banea *et al.*, 2012] propose a system based on WordNet similarity measures, semantic models (such as LSA, and ESA), and dependency graph alignment. In an attempt to compute dependency graph alignment scores they used the method based on [Mohler and Mihalcea, 2009], which computes the score in two steps. First, it assigns similarity scores for each node in the dependency graphs of the two sentences. Then, it finds the optimal alignment between dependency nodes in two graphs. This optimality is determined by a learning algorithm which computes the similarity score between pair of nodes and their corresponding subgraphs.

A rule-based system was proposed by [AbdelRahman and Blake, 2012]. The Sbdlrhmn system branches from the above in its nature. It is a rule-based system that tags *head-nouns*, *main-verbs*, and *named-entities* as important information in each sentence. These preferred words are assigned a

[11] Given a T-H pair, the BIUTEE system recognizes whether the hypothesis (H) can be inferred from the text (T). It applies a sequence of knowledge-based transformations, by which the text is transformed into the hypothesis. The system decides whether the text entails the hypothesis by observing the quality of this sequence.

score if they co-occur inter-sentences. Then, as in TakeLab, the system extracts predicate structures of type *subject-verb-object* for every verb in both sentences and sets about aligning them. If predicates align based on word overlap, they are given a score. Otherwise, WordNet (*lesk* metric) is used to measure the similarity of (and assign a score to) the particular unaligned predicate pairs. Finally, a series of *if-then* rules based on the combinations of scores decides the level of semantic relatedness between two input sentences.

In [Souza *et al.*, 2012] the authors explore two classes of measures, word similarity metrics and machine translation metrics. They use the YAGO2[12][Hoffart *et al.*, 2011] semantic knowledge base which was constructed from Wikipedia, WordNet and Geonames[13]. For each entry, YAGO2 gives a set of relations between the input entry and the entries in its knowledge base. They also use LSA and n-gram models, and incorporate machine translation error measures such as TER (translation error rate), and WER (word error rate). TER measures the number of edits required to change a system output into one of the references. WER is based on the Levenstein distance but, as opposed to Levenstein distance that works on character level, works on word level and calculates the number of words that differ between a machine generated translation and a human translation. They also extract syntactical features based on dependency parsing, POS tags, and phrase chunks. However, according to their result, their best performance was achieved by using only LSA and features based on n-grams and edit distance metrics.

[Islam and Inkpen, 2009] propose a system that can measure similarity between sentences or paragraphs. They introduce a modified version of *longest common subsequence*, which relaxes it, by letting the subsequence starts either at first character or any character of words. On corpus level, they also use Second Order Co-occurrence PMI (SOCPMI) that I've discussed earlier in "Others" Section. Having these similarity scores on word-to-word level, they use a method for propagating it to sentence-to-sentence similarity level that finds the most similar match for each word and then sum up these maximum similarity scores.

[12]http://www.mpi-inf.mpg.de/yago-naga/yago/
[13]http://www.geonames.org

Chapter 3

Experimental Setup

3.1 Datasets

I used STS-2012 dataset's training and test sets as the main dataset in this work. Also, as mentioned earlier, I participated in STS-2013 shared task and therefore I will talk about the 2013 dataset and its characteristics as well.

3.1.1 STS-2012 Datasets

The STS-2012 dataset's training and test sets composed of three and five datasets, see Table 4. Each set contains pairs of sentences and the goal is to measure the similarity between sentences in each pair. This measurement's output can vary within an interval [0,5], in which 5 means the two sentences are completely equivalent, and 0 means they are on different topics. The gold standard was assembled using mechanical turks, gathering 5 scores per sentence pair. Amazon mechanical turks are workers to complete tasks that computers are currently unable to do. The gold standard is the average of these 5 scores.

Training Datasets	
MSR-Paraphrase, Microsoft Research Paraphrase Corpus	750 pairs
MSR-Video, Microsoft Research Video Description Corpus	750 pairs
SMTeuroparl: WMT2008 develoment dataset (Europarl section)	734 pairs
Test Datasets	
MSR-Paraphrase, Microsoft Research Paraphrase Corpus	750 pairs
MSR-Video, Microsoft Research Video Description Corpus	750 pairs
SMTeuroparl: WMT2008 development dataset (Europarl section)	459 pairs
SMTnews: news conversation sentence pairs from WMT	399 pairs
OnWN: the first sentence comes from Ontonotes and the second from WordNet	750 pairs

Table 4: SemEval-2012 Semantic Textual Similarity (STS) shared task's training and test datasets

The training set consists of 3 different datasets of text which were gathered from Microsoft Paraphrase corpus (MSR-Paraphrase), Microsoft Video Description Corpus (MSR-Video), and European Parliaments corpus (SMTeuroparl).

	# of VP	# of NP	Avg. Length
MSR-Video	2833	3478	8
MSR-Paraphrase	4931	11807	22
SMTeuroparl	7769	15499	30

Table 5: Statistical facts - SemEval-2012 STS sharedTask's training set

Regarding the statistical characteristics of each of these datasets we collected some facts that are presented in Table 5. Microsoft Video Description (MSR-Video) dataset contains shorter and simpler (according to the statistical facts) sentences. In fact, the creation process of the MSR-Video dataset supports these numerical facts. Because the corpus was collected by showing a segment of a YouTube video to mechanical turks and asking them to give a one-sentence description of the main event in the video. An example of a pair of sentences from MSR-Video set in the STS-2012 training set:

(3.1) (a) A big turtle is walking.

(b) The tortoise is walking.

The original corpus of Microsoft Paraphrase dataset (MSR-Paraphrase) was collected over a period of 18 months from online news sources. Then each pair was shown to 2 human judges to give a binary judgment wether the two sentences in the pair are semantically equivalent. Disagreements were resolved by a 3rd judge. Note that pairs with a word-based Levenshtein distance less than 0.8 are

already excluded from the MSR-Paraphrase dataset. Also, in comparison with MSR-Video, MSR-Paraphrase contains longer sentences which naturally carry more verb phrases and noun phrases as well. In contrast with MSR-Video dataset, which was artificially created, MSR-Paraphrase contains sentences which are more real in terms of lexical and syntactic complexity. An example of a pair of sentences from MSR-Paraphrase set in the STS-2012 trainig set:

(3.2) (a) *Semiconductor giant Intel Corp. said yesterday that its second-quarter profits doubled from a year ago as stronger-than-expected demand for computer microprocessors offset the weakness of its communications chip business.*

(b) *Intel Corp.'s second-quarter profits doubled and revenues grew 8 percent from a year ago as the chip-making giant reported stronger-than-expected demand for personal computer microprocessors..*

The last dataset in the training sets, SMTeuroparl, was collected from the development dataset of "Automatic Evaluation of Machine Translation" shared task. This shared task was part of the "Workshop of Machine Translation" 2008 (WMT2008) [Callison-Burch *et al.*, 2008]. The participants were asked to develop an evaluation metric for measuring the quality of the translations submitted to the other shared tasks of the same workshop, called "Machine Translation for European Languages". As Table 5 shows it has the highest number of verb phrases, and noun phrases and the longest sentences among the other datasets. An example of a pair of sentences from SMTeuroparl set in the STS-2012 trainig set:

(3.3) (a) *While withdrawing the resolution, however, I express the conviction that this Parliament would have made its voice heard better if, in anticipation of the Council of Nice, it had devoted a specific and separate resolution to these important and difficult topics of the Intergovernmental Conference, instead of dealing with them in a single resolution that also embraces all the other points on the Council agenda.*

(b) *By withdrawing our resolution, I express however the conviction that our Parliament*

would have better done to hear its voice while devoting, for Conseil of Nice, a resolution specific and distinct with the so important and so difficult topics to the intergovernmental Conference, rather than to treat them within the framework of one only resolution also including all the other points of about an agenda of the Council.

The STS-2013 test set contains 5 datasets of text, three of which were collected from the same resources as the training sets. The other two test sets, OnWN and SMTnews, are given as surprise sets to examine the reliability of proposed systems in dealing with unforeseen data. The OnWN dataset comprises 750 pairs of glosses from OntoNotes[1] and WordNet senses. Half of these pairs were collected from senses that were recognized to be equivalent and the remaining from disparate senses. An example of a pair of sentences from OnWN set in the STS-2012 test set:

(3.4) (a) *The release of pressure built up during a stop consonant.*

(b) *the terminal forced release of pressure built up during the occlusive phase of a stop consonant.*

SMTnews dataset contains 399 sentence pairs from translation systems submitted to WMT2007 and ranked by human. An example of a pair of sentences from SMTnews set in the STS-2012 test set:

(3.5) (a) *Only a month ago, Mubarak dismissed demands for constitutional reform as "futile."*

(b) *Only a month before, muybarak refused the demands of constitutional reform by taxing them with "futile.*

We did not participate in the STS shared task-2012 but we wanted to simulate the same procedures the participants followed, we only used the test sets for the testing purpose and analysis after the experiments. During the training phase, the only data that was considered was the training sets of the shared task.

[1] "The OntoNotes project focuses on a domain independent representation of literal meaning that includes predicate structure, word sense, ontology linking, and coreference" [Hovy *et al.*, 2006].

3.1.2 STS-2013 Datasets

In addition to STS-2012 datasets, I participated in STS-2013 shared task and therefore used this recent dataset for the evaluation purpose. As it is demonstrated in Table 6, the STS-2013 task uses all the training and test data from STS-2012 as the training sets and introduces 4 test sets, 2 (FNWN, and headlines) of which are surprise sets.

Training Datasets	
MSR-Paraphrase, Microsoft Research Paraphrase Corpus	1500 pairs
MSR-Video, Microsoft Research Video Description Corpus	1500 pairs
SMTeuroparl: WMT2008 develoment dataset	1193 pairs
SMTnews: news conversation sentence pairs from WMT	399 pairs
OnWN: definitions from Ontonotes and WordNet	750 pairs
Test Datasets	
OnWN: definitions from Ontonotes and WordNet	561 pairs
FNWN: sense definitions from WordNet and FrameNet	189 pairs
SMT: SMT dataset comes from DARPA GALE HTER and HyTER	750 pairs
headlines: headlines mined by European Media Monitor	750 pairs

Table 6: Semantic Textual Similarity (STS) 2013 shared task's training and test datasets

The sentences in FNWN test set are sense definitions from WordNet and FrameNet. An interesting consideration here is the fact that some FrameNet definitions involve more than one sentence. An example of a pair of sentences from FNWN set in the STS-2013 test set:

(3.6) (a) *this frame has to do with scientific taxonomy. the lexical units in this frame include the seven major classifications in which organisms are grouped based on common biological characteristics. these classifications are labeled rank in this frame. the members or subtype of the rank is often expressed with the use of the target-denoting noun.*

(b) *(biology) taxonomic group containing one or more families*

The SMT dataset of the 2013 shared task, was selected from DARPA GALE HTER and HyTER datasets, where one sentence is a MT output and the other is a reference translation where a reference is generated based on human post editing (provided by LDC) or an original human reference (provided by LDC) or a human generated reference based on FSM as described in (Dreyer and Marcu, NAACL 2012). The reference comes from post edited translations. An example of a pair of sentences from SMT set in the STS-2013 test set:

(3.7) (a) *Neither the studies you chose please you, nor the profession you practiced pleases you, ... nor the wife you loved has made you happy, and nor the dreams you hanged on to have materialized?*

(b) *neither the study you chose was liked by you , or you are satisfied with the business which you pursued , nor you are happy with the partner you cherish , nor were the ambitions you held on to accomplished .*

The headlines test set are gathered using headlines mined from several news sources by European Media Monitor using the RSS feed[2]. An example of a pair of sentences from headlines set in the STS-2013 test set:

(3.8) (a) *Drug lord captured by marines in Mexico*

(b) *Suspected drug lord known as El Taliban held in Mexico*

I will talk about the results and analysis of the results of STS-2013 shared task in Sections 5, and 6. However, during the rest of this chapter the data that is used is the data from STS-2012 shared task.

3.2 Features

Before extracting the features the following preprocessing steps are required:

tokenizing annotating input texts' tokens

lemmatizing identifying lemma for each tokens. Lemma is the canonical form of a set of words. For example, "write", "writes", "wrote" and "writing" are forms of the same lexeme, with "write" as the lemma.

sentence splitting annotating input texts' sentences

part of speech (POS) tagging adding POS tags to every tokens of the input texts.

[2]http://emm.newsexplorer.eu/NewsExplorer/home/en/latest.html

for preprocessing steps General Architecture for Text Engineering (GATE)[3] [Cunningham *et al.*, 2011], an open source software for developing resources that process text, was mostly used. Also, I performed stop word removal only for calculating ESA and WordNet-based measures. For the rest of the features, stop words are not discarded.

After the preprocessing steps, I extract five categories of lexical features: string similarity [Cohen *et al.*, 2003], n-grams [Lin and Och, 2004], WordNet distance [Budanitsky and Hirst, 2006], Roget's Thesaurus [Jarmasz and Szpakowicz, 2003], and ESA [Gabrilovich and Markovitch, 2007]. My intuition was based on the literature study, that shows that these methods and resources provide a robust and stable performance in any similarity-related experiments. I also wanted to examine Roget's Thesaurus, since it was rarely used by the researchers for measuring similarity (we will see that in fact Roget's is almost as powerful as WordNet and yet it has way lower word coverage compared to WordNet.)

The following sections discuss the individual features used in each category. To assess their usefulness, single-feature models are trained using cross-validation on the training data, it is these results that are reported in the tables presented throughout Section 3.2. All performance tables also include a model where all features are trained together (marked with a star) and the results for testing on all the different datasets together (last row, captioned ALL).

3.2.1 String Similarity Metrics

In order to calculate similarity for a given pair, 5 common similarity/distance measures were used: Longest Common Substring, Longest Common Subsequence, Jaro, Jaro-Winkler, Monge-Elkan, and ROUGE-W (denoted by RO-W). We also added a feature that counts normalized lemma overlap here, even though it is not strictly a string-based technique.

[3] http://gate.ac.uk

	lemma	jar	jarWk	lcsubsq	lcsubst	monElk	RO-W	str*
MSR-Video	0.36	0.19	-0.07	0.48	0.33	0.14	0.75	0.79
MSR-Paraphrase	0.39	0.33	0.17	0.18	0.20	0.54	0.52	0.64
SMTeuroparl	0.55	0.54	0.41	0.30	0.25	0.38	0.53	0.74
ALL	0.41	0.28	0.18	0.22	0.13	0.38	0.62	0.76

Table 7: Performance of string-based features, "str*" represents all of the string-based measures being used together.

Table 7 shows that interestingly, the model based on ROUGE-W showed a strong, on average (weighted average), correlation factor of 0.6, which is the highest performing feature in this group. The negative value mean that there is a negative correlation between the scores predicted by the system trained with "jarWk" as the only feature and the gold standard values.

3.2.2 N-grams Models

I used the ROUGE package, originally developed for automated evaluation of summaries [Lin and Och, 2004], to extract n-gram models. As reported in [Lin, 2004], the most effective n-gram models for measuring the similarity between small text fragments are:

- **ROUGE-1**, based on Unigrams
- **ROUGE-2**, based on Bigrams
- **ROUGE-SU4**, based on 4-Skip bigrams (including Unigrams)

	ROUGE-1	ROUGE-2	ROUGE-SU4	ROUGE*
MSR-Video	0.78	0.62	0.67	0.81
MSR-Paraphrase	0.63	0.41	0.56	0.63
SMTeuroparl	0.58	0.39	0.46	0.58
ALL	0.71	0.60	0.61	0.75

Table 8: Performance of ROUGE-based features. "ROUGE*" represents ROUGE-1, 2, SU4 being used together.

As expected, n-grams are effective in calculating the similarity of shorter fragment of texts, while their performance decreases on measuring similarity of longer texts.

3.2.3 WordNet-based Measures

To address the implicit similarity between sentences in each pair, two WordNet-based metrics, Lin [Lin, 1998b] and Jiang-Conrath [Jiang and Conrath, 1997] are used. A evaluation study of WordNet-based measures, conducted by Hirst [Budanitsky and Hirst, 2006], shows that Lin and Jiang-Conrath metrics are the most reliable measures for similarity measurement across different tests. In fact, both of these measures shown the highest correlation, on average, with the human ratings of similarity in Miler and Charles [Miller and Charles, 1991], and Rubenstein and Goodenough [Rubenstein and Goodenough, 1965] word pairs. The WordNet::Similarity package [Pedersen *et al.*, 2004] was used for extracting these two scores.

Also, in order to scale the word-to-word similarity produced by these metrics to sentence-to-sentence level, the method previously introduced by Mihalcea [Mihalcea *et al.*, 2006] was applied. This method, for a given similarity metric, for each word w in sentence S_1 finds the word from sentence S_2 which gives the maximum similarity score, $maxSim(w, S_2)$. Then for each calculated score it weights the score by incorporating the *idf* weight of its corresponding word. In order to get these *idf* weights, I provided a word list by extracting all words from WordNet, then used the *Sketch Engine*[4] [Kilgarriff *et al.*, 2004] (a corpus query system that provides different types of summaries of words behavior.) and the *British National Corpus* as the resource for calculating *idf*. Having these *idf* weights and the raw maximum score for each word in S_1, I sum up these scores and normalize them. The same procedure is applied to S_2 and the final score is calculated by taking the average:

$$sim(S_1, S_2) = \frac{1}{2}\left(\frac{\sum_{w\in\{S_1\}}(maxSim(w, S_2) * idf(w))}{\sum_{w\in\{S_1\}} idf(w)} + \frac{\sum_{w\in\{S_2\}}(maxSim(w, S_1) * idf(w))}{\sum_{w\in\{S_2\}} idf(w)}\right)$$

For example, consider the following pair:

(3.9) (a) The woman is playing the violin.

[4] http://www.sketchengine.co.uk/

(b) The young lady enjoys listening to the guitar.

Starting with each of the two sentences (ignoring stop words), I determine the most similar word in the other text segment, according to the Jiang-Conrath(jcn) similarity measure. Next, using Mihalcea's method I combine the word similarities and their corresponding *idf*, and determine the semantic similarity of the two sentences (the left side of the table starts from sentence a, while the right side starts from sentence b):

Sentence a	Sentence b	$maxSim_{jcn}$	*idf*	Sentence b	Sentence a	$maxSim_{jcn}$	*idf*
woman	lady	0.35	0.31	young	-	0	0.17
playing	enjoys	0.09	0.32	lady	woman	0.35	0.50
violin	guitar	0.25	1.33	enjoys	playing	0.09	0.94
				listening	playing	0.077	0.50
				guitar	violin	0.25	1.26

Table 9: WordNet similarity scores based on jcn and *idf*

Note that the word "young" is an adjective and because the first sentence does not have any adjective the corresponding entry in Table 9 is left empty. By plugging in the numbers in Table 9 into Mihalcea's method, the similarity between these two sentences is 0.21.

3.2.4 Roget's Thesaurus-based Measure

Roget's Thesaurus provides more links between different word types than WordNet, for example a noun *"bank"* is connected to a verb *"invest"* by determining the common head *"Lending"*. As it was mentioned earlier, the distance of two terms decreases within the interval of [0,16], as the the common head that subsumes them moves from top to the bottom and becomes more specific. The electronic version of Roget's Thesaurus [Jarmasz and Szpakowicz, 2003] is used for extracting this score.

	jcn	lin	Roget's	KB*	ESA
MSR-Video	0.75	0.71	0.72	0.78	0.83
MSR-Paraphrase	0.26	0.24	0.20	0.27	0.33
SMTeuroparl	0.21	0.21	0.21	0.50	0.45
ALL	0.58	0.58	0.68	0.71	0.73

Table 10: Performance of features based on WordNet, Roget's, and ESA. "KB*" represents jcn, lin, Roget's being used together. ESA see below.

The results in Table 10 indicate that, Jiang-Conrath (denoted by jcn) performs reliably across different datasets. Surprisingly, despite the small size of Roget's coverage compared to WordNet, in the ablation test it demonstrated a strong contribution to the systems performance. Roget's has correlation scores identical or very close to Lin (denoted by lin), and in the SMTeuroparl training subset it achieves the same performance as Jiang-Conrath. When we combined all the training subsets, Roget's outperformed both Lin and Jiang-Conrath by a margin of 10% with a correlation factor of 0.68.

3.2.5 Explicit Semantic Model

In order to have broader coverage on word types not represented in lexical resources, specifically for named entities, we add ESA generated features to our feature space. The results are shown in the last column of Table 10. As the table showed, the average correlation factor achieved by ESA across the training sets was 0.53, which is 2% above the performance reported on other knowledge-based metrics.

3.2.6 Ablation Assessment

The single feature models we compared led to exclude *Jaro-Winkler*, *longest common substring*, and *Monge Elkan* from the feature space and we consider the remaining features to be effective. Because the correlation between the scores predicted by the models trained with these single features and the gold standard values (on training set) were negative or did not show stability across different datasets in the training set. Therefore, the corresponding features were identified as poor and hence got excluded from the feature space.

3.3 Apparatus

Since the output of the system should be distributed within the interval [0,5] and regression provides this continuity, we tested linear regression, logistic regression and support vector regression

(SVR) [Smola and Schölkopf, 2004] using the training set and based on their performance on the training set chose SVR as the learning algorithm. The Java API of Weka [Hall *et al.*, 2009] and its implementation of Support Vector Regressor (SVR) algorithm called Sequential Minimal Optimization regressor (*SMOreg*) was used for this purpose.

The goal of the SVM regressor, similar to SVM, is to estimate a function $f(x)$ that is as close as possible to the target value d_i for every x_i in the training set and at the same time, does not care about errors as long as they are less than ε, but does not accept any deviation (between the predicted and the target values) larger than this ε value [Shevade *et al.*, 2000]. In fact, SVM ignores training instances that lie beyond the margin. This results in the main difference between Support Vector Regressor and other types of regressor, which is the fact that Support Vector Regressor builds models based on a subset (called support vectors) of the the training set and not the full training set which makes it very fast and efficient in training step.

In this chapter I introduced the datasets, the pruned feature space, and the learning algorithm that I used in my experiments. In the following chapter, I go a step further and explain the two phase supervised learning method that I proposed for combining these features.

Chapter 4

Two Phase Supervised Learning

4.1 Methodology

We propose a two phase supervised learning method for the STS shared task. As Figure 4 demonstrates, to train the two phase model we first generate all combinations of the features extracted (*jaro*, *Lemma*, *lcs*, *ROUGE-W*, *ROUGE-1*, *ROUGE-2*, *ROUGE-SU4*, *roget*, *lin*, *jcn*, *ESA*) via our pipeline (*Basic Features*). In *Preprocessing* step for a feature space of size N we generate $2^N - 1$ non-empty combinations. Since we have 11 features, the preprocessing step generates 2047 combinations. Then in *Two Phase Model Training* step, for each combination as is shown in the figure, a separate Support Vector Regressor (SVR), called *Phase One Model* is trained. Each *Phase One Model* for instances from the training set, predicts a score shown by ps_i, where i stands for the index of the corresponding *Phase One Model*. Therefore for each instance in the training set we get $2^N - 1$ predicted scores. These $2^N - 1$ predicted scores form a new feature vector called *Phase Two Features*. So for each instance in the training set a feature vector of size $2^N - 1$ will be created. Finally a second level SVR, called *Phase Two Model* is trained using the training set and this new feature space.

Once the training step is done, we have two types of trained SVRs: $2^N - 1$ *Phase One Models*

which, as we explained, are directly based on *Basic Features*, and a *Phase Two Model*, which is trained on *Phase Two Features*.

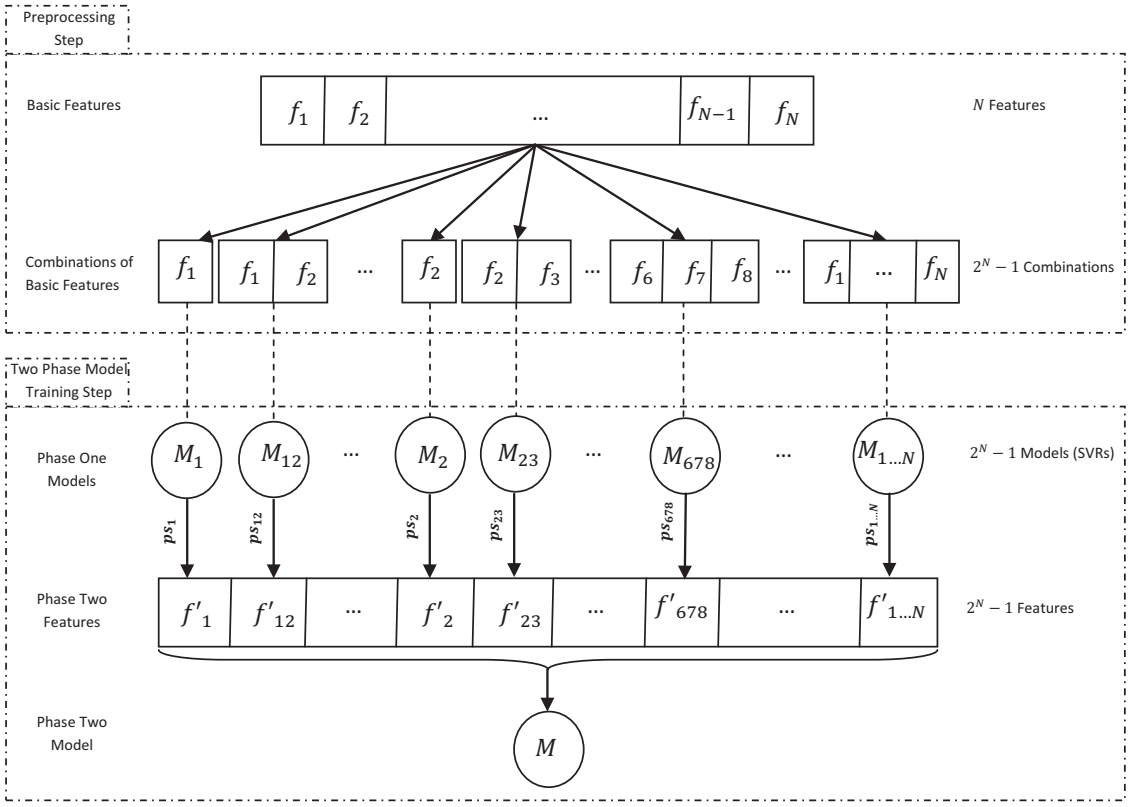


Figure 4: Two phase supervised learning method's architecture

Then for any sentence pair in the test set we proceed as follow: First we send it as an input to the *Phase One Models* and get 2047 predicted scores. Using these scores from *Phase One Models*, *Phase Two Features* are created. This new feature vector then feeds into a *Phase Two Model* and it predicts the final score.

4.2 Implementation

In this section I present the pseudocodes for the required procedures and modules I implemented.

The abstract overview of the flow of the proposed system is as follows:

```
/*OVER TRAINING SETS*/
Run Pipeline                    /*extracts Basic Features*/
Generate Subsets                /*generates non-empty subsets of Basic Features*/
Train Phase One Models          /*trains a phase one model based on each subset*/
Build Phase Two Feature Space   /*creates the phase two feature space*/
Train Phase Two Model           /*creates the final model*/
```

In below, I demonstrate the way that *Run Pipeline* functions:

```
/*Modules are wrapped in GATE environment.*/
/*Modules used from GATE plug ins are marked with @G sign.*/
Convert To XML             /*converts raw data to XML*/
ANNIE Tokenizer @G         /*tokenizes the text*/
Sentence Splitter          /*annotates Sentences*/
POS Tagger @G              /*adds POS tags*/
Morphological Analyzer @G  /*adds lemma of each token*/
String Similarity          /*calculates string similarity scores*/
ROUGE                      /*calculates ROUGE-based scores*/
Explicit Semantic Analyzer /*calculates ESA-based scores*/
WordNet Similarity         /*calculates WordNet-based scores*/
Roget's Similarity         /*calculates Roget's-based scores */
Export To Arff             /*exports generated scores to .arff format*/
```

After these steps, all of the input pairs of sentences are represented by these extracted scores. Therefore, sentences are converted to vectors of numbers. I refer to each of these vectors as feature vector. For example for the pair:

(4.1) (a) A woman and man are dancing in the rain.

(b) A man and woman are dancing in rain.

this is the systems output in XML format for the above example:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <pair p_id="2" gs="5.0" task="STS" esa="1.0" lin="1.0" jcn="1.0" roget="0.9"
  lemma="0.94" jaro="0.92" lcs="0.94" ROUGE-1="1.0" ROUGE-2="0.33" ROUGE-W="0.58"
  ROUGE-SU4="0.88">
3   <text t_id="1">A woman and man are dancing in the rain.</text>
4   <text t_id="2">A man and woman are dancing in rain.</text>
5 </pair>
```

and its corresponding arff data ready to use with Weka is:

```
@ RELATION     "Semantic Textual Similarity"

@ ATTRIBUTE    esa              REAL
@ ATTRIBUTE    lin              REAL
@ ATTRIBUTE    jcn              REAL
@ ATTRIBUTE    roget            REAL
@ ATTRIBUTE    lemma            REAL
@ ATTRIBUTE    jaro             REAL
@ ATTRIBUTE    lcs              REAL
@ ATTRIBUTE    ROUGE-1          REAL
@ ATTRIBUTE    ROUGE-2          REAL
@ ATTRIBUTE    ROUGE-W          REAL
@ ATTRIBUTE    ROUGE-SU4        REAL
@ ATTRIBUTE    gs               REAL

@ DATA
1.0 1.0 1.0 0.9 0.94 0.92 0.94 1.0 0.33 0.58 0.88 5.0
```

Features are defined with "@ATTRIBUTE", their name, and their type, and feature vectors are placed under the"@DATA", each row presenting a feature vector.

Generate Subsets produces all possible non-empty subsets. Each subset contains a number(s)

standing for the id of the feature it is representing. For example in $subsets_k = \{1, 7, 9\}$, 1 stands for *esa* feature, 7 stands for *lcs* feature, and 9 stands for the feature calculated by *ROUGE-2* metric.

Algorithm 1 demonstrates the followed procedures for building *Phase One Models*.

Algorithm 1 Train Phase One Models

```
1: for all trainingSet ∈ trainingSets do
2:   for all subset ∈ subsets do
3:     filteredtrainingSet ← filter(subset, trainingSet)
4:     PhaseOneModel ← train(filteredtrainingSet, SVR)
5:     store(PhaseOneModel)
6:   end for
7: end for
```

Note that in line 3, the *filter* function, takes a subset of the features and a training set, keeps all the features that are specified by the *subset* and excludes all the remaining features and returns a new training set called *filteredtrainingSet*. In line 4, this new training set is used to train a SVR (*Phase One Model*).

Algorithm 2 shows the flow of commands that should be executed to build the new feature space called *Build Phase Two Feature Space*.

Algorithm 2 Build Phase Two Feature Space

```
1: for all trainingSet ∈ trainingSets do
2:   for all pair ∈ trainingSet do
3:     for all POModel ∈ PhaseOneModels do
4:       phaseTwoFeatureSpace_{pair_id,POModel_id} ← predict(pair, POModel)
5:     end for
6:   end for
7:   overwrite(trainingSet, phaseTwoFeatureSpace)
8: end for
```

In line 4, the trained model *POModel* predicts a score for the specified input *pair*. At each iteration of the *for-loop* (lines 3 to 5), a cell at index *pair_id* of the new feature vector shown by $phaseTwoFeatureSpace_{pair\_id,POModel\_id}$ is being filled. Once this *for-loop* is finished, we have a new feature vector for the *pair* with the id *p_id*.

Within the body of the *for-loop* (lines 2 to 6) a new feature vector is being created for every pair in the training set. In line 7, the generated *phaseTwoFeatureSpace* overwrites the *Basic Feature Space* in the specified *trainingSet*. Therefore, by the end of Algorithm 2 training sets are represented

in the new feature space called *Phase Two Feature Space*.

Finally in Algorithm 3, using the *Phase Two Feature Space* the final model called *Phase Two Model* is being trained.

Algorithm 3 Train Phase Two Model

1: **for all** $trainingSet \in trainingSets$ **do**
2: $PTModel \leftarrow train(trainingSet, SVR)$
3: **end for**

Once the training step is finished, the trained models are ready for the test phase and I proceed as follows:

```
/*OVER TEST SETS*/

Run Pipeline                    /*extracts Basic Features*/

Build Phase Two Feature Space   /*creates a new feature space*/

Test Phase Two Model            /*predicts the final output of the system*/
```

The *Run Pipeline* and the *Build Phase Two Feature Space* are the same as I explained before. The only difference is that during the test, instead of training sets, test sets are being used. Algorithm 4 demonstrates the procedures of the testing step.

Algorithm 4 Test Phase Two Model

1: **for all** $testSet \in tesSets$ **do**
2: **for all** $pair \in tesSet$ **do**
3: $predictedScore_{pair} \leftarrow predict(pair, PTModel)$
4: **end for**
5: **end for**

In line 3 the trained phase two model, represented by $PTModel$, predicts a score for the specified $pair$. This predicted score is the final output of our system.

4.3 Discussion

In general, I suggest the use of the proposed method when there is no obvious generic solution for a task. This means that if we are already aware of the existence of a feature set which will provide

the best performance, there is no need to use our proposed method. Unfortunately, in almost all cases, this is not the situation. Also, we are dealing with datasets that were gathered from different resources each of which having their own characteristics. Therefore, using a single subset of the feature set, or a single possibility of combining features (i.e. by using all of the features together) for training a classifier might not work across all of these different datasets. In this section I will try to address some of the discussions that may raise regarding the proposed method.

Time and Resources : On a machine with "2 Intel(R) Xeon(R) CPU E5645", 96 GB of RAM, and with multi-threading (thread pool of size 200, a training process was assigned to each thread) it roughly took 5.5 hours to train 2047 *Phase One Models* using 2234 pairs in the training sets and another 7.5 hours to build *Phase Two Feature Space* for the training data. Building the *Phase Two Feature Space* for the test sets took roughly 10 hours for 3108 pairs in the test sets. Evidently, our method is more resource intensive compared to single-run learning methods. However in Chapter 5, I show that this computational complexity is traded off for achieving better results compared to the model trained using standard single-run learning.

Comparison to Traditional Feature Selection Our intuition for not applying any feature selection methods, was to not remove features based on the training set data. For sure, removing irrelevant features is a crucial step and we did the ablation assessment. In fact, the *Phase Two Model* of our proposed approach, learns the relationship between each *Phase One Model's* prediction and the gold standard. This is beneficial because a *Phase One Model* might consistently make the correct predictions for a particular class, while another model may perform poorly on that class. Besides, the feature selection is based on training set which increases the risk of overfitting, while performing the *Selection* process is also a computationally expensive process (still less than our method's computational expense).

Since our method differs from the other methods, especially those that use *feature selection* techniques, in terms of its approach in dealing with features, in Chapter 5, I conduct a separate experiment in order to compare our method, on STS task, with a single-run learning method that uses

the feature space selected by two of the well-known *feature selection* algorithms: *Exhaustive Search*, and *Forward Feature Selection*.

The *Exhaustive Search* feature selection tries all different possibilities of feature subsets and selects the one that provides the best performance on training set. While the *Forward Feature Selection*, begins by evaluating all feature subsets of size one, then it finds the best subsets of two features from the subsets containing f_i, the best single feature selected in the previous step, and all the remaining features, and so on. The greediness of this method is also problematic: For example if f_1 is the best single feature, it does not guarantee that either $\{X_1,X_2\}$ or $\{X_1,X_3\}$ would perform better than $\{X_2,X_3\}$ [Guyon and Elisseeff, 2003].

In this chapter, I introduced the proposed method with some technical details. In the next chapter, I will explain the experiments that I conducted and their results. I will compare the results and show that the best performance is achieved by the proposed method.

Chapter 5

Experiments

In order to assess our proposed method and features, three types of experiments were conducted using: *Standard Learning*, *Standard Learning with Feature Selection*, and *Two Phase Learning*:

- **Standard Learning** The regressor was trained on the training set with all 11 *Basic Features* and tested on the test sets.

- **Standard Learning with Feature Selection** *feature selection* is applied before performing the standard learning. I used two of the well-known algorithms, *Exhaustive Search*, and *Forward Feature Selection*. Both *Feature Selection* methods resulted in the same feature subset which consists of 3 features : esa, lin, ROUGE-1. The regressor was trained on the training set with these three features and tested on the test sets.

- **Two Phase Learning** First, all combinations of the *Basic Features* were generated, which are 2047 combinations. Then using these feature combinations, 2047 *Phase One Models* were trained. Further, each instance in the training set was used as an input and a score was predicted by each of these *Phase One Models*. This results in having 2047 predicted scores per instance in the training set. These predicted scores form *Phase Two Features*. Finally, the *Phase Two Model* is trained using the training set on this new feature space.

5.1 Definition of Evaluation Metric

Following the definition of task and for the purpose of evaluating the performance, Pearson Correlation Coefficient (also called correlation factor/coefficient) is used. It measures the correlation (linear dependence) between two variables. For example the correlation between variables x and y is calculated as follow:

$$r = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{[n \sum x_i^2 - (\sum x_i)^2][n \sum y_i^2 - (\sum y_i)^2]}}$$

where n is the size of the sample.

In case of the STS shared task-2012 the correlation is calculated between the gold standard values and the predicted values. This correlation score varies within the interval of [-1,1], while -1 implies that all data points lie on a line for which one variable decreases as the other one increases and 1 implies that a linear equation describes the relationship between the two variables perfectly.

5.2 Preliminary Evaluation

Because of the continuity of the output space, the participants in the STS shared task-2012 were asked to use correlation coefficient as their evaluation metric. Therefore, I used the same metric for evaluating my system. The result of my experiments are provided in Table 11. Compared to the *Standard Learning* and *Standard Learning with Feature Selection*, in all cases the *Two Phase Learning* achieved better results with the margin of 2% to 7%. The last row shows the result achieved by the baseline system, using the cosine similarity between the token vectors of sentences in each pair, provided by the shared task organizers [Agirre *et al.*, 2012].

	MSR-Paraphrase	MSR-Video	SMTeuroparl	SMTnews	OnWN
Standard	0.65	0.87	0.51	0.36	0.66
Feature Selection	0.59	0.87	0.52	0.40	0.67
Two Phase	0.67	0.89	0.56	0.43	0.72
UKP-run2 (best)	0.68	0.87	0.52	0.49	0.66
STS-2012 baseline	0.43	0.29	0.45	0.39	0.58

Table 11: Correlation Coefficients for the Standard Learning, Standard Learning with Feature Selection and the Two Phase Learning method experiments

In order to put our results into perspective, the reported results of the submitted systems to the STS shared task-2012 were taken into consideration. In comparison with the UKP-run2 system, our system ranked 1st in MSR-Video and SMT-europarl, OnWN test sets. In MSR-Paraphrase with less than 1% margin, our system was beaten by UKP-run2 and in the last test set, STMnews, our system had a poor performance and was beaten by the margin of 5%. Overall our system on average outperformed all the 89 submitted systems (including the baseline) to the task and achieved the average correlation coefficient of 0.69 which is 2% higher than the best performing system.

Also, as it can be seen the *Standard Learning with Feature Selection* performs almost equally or better and in one case (on MSR-Paraphrase test set) worse than *Standard Learning* method. However, as it can be seen, its results are still below the ones achieved by our proposed method.

5.3 Statistical Significance

Finding if something is a chance event or not is one of the classical problems of statistics which is referred to as *Hypothesis Testing* [Manning and Schütze, 1999]. Given an event, a test of statistical significance first formulates a *null hypothesis* H_0 that the event is a chance event and then it computes the probability p that the event would occur if H_0 was true. It rejects the *null hypothesis* if p is too low, for example below a predefined *significance level* of 0.05. In simple words, a *statistical significance test* looks at the statistical facts and at the same time considers the number of samples that we have seen. Therefore, even if we have found an interesting pattern, it will reject it if we have not seen enough data.

In this work, a statistical significance test can be applied for two purposes: First, to see if the calculated correlation between the gold standard and the generated outputs is statistically significant. Second, to see if the improvements I obtained by using the *Two Phase Learning*, compared to the other methods, are statistically significant.

In order to conduct a *statistical significance test*, one must have access to the output of the two compared systems. Even though I was willing to conduct this test on the output of our system and the output of the best performed system of the STS shared task-2012, their outputs were not available [Agirre *et al.*, 2012].

5.3.1 Correlations Significance

Once I have calculated correlation coefficients, I can perform the statistical significance test to see whether the achieved correlation occurred by chance or not (is a real correlation or not). In this case, I am testing against the *null hypothesis* that $r = 0$ (r stands for correlation coefficient), which means there is no correlation, neither positive nor negative, between the gold standard and the predicted outputs.

I used the common significance level of $\alpha = 0.05$ and also need to use the corresponding degrees of freedom, df, which is equal to $N - 2$, where N is the number of samples. The number of degrees of freedom is the number of independent pieces of data being used to make a calculation. It can be viewed as the number of independent parameters available to fit a model to data. Since I had no strong prior theory to suggest whether the relationship between predicted scores and the gold standards would be positive or negative, I used two-tailed test. Having these parameters set, I can now test the significance of the correlation I found.

	MSR-Paraphrase	MSR-Video	SMTeuroparl	SMTnews	OnWN
Test Set's Size - N	750	750	459	399	750
Significance Values	0.088	0.088	0.098	0.113	0.088

Table 12: Significance values of r, with $\alpha = 0.05$.

Table 12 presents the critical values for each test set. For example on first row, for the test set

MSR-Paraphrase the significant value is 0.088 which means that if the correlation is greater 0.088 or less than -0.088 (this is a two-tailed test) I can conclude that the probability that the correlations on Table 11 for this test set are by chance is less than 0.05. Since the achieved correlations by *Standard Learning*, *Standard Learning with Feature Selection*, and *Two Phase Learning* are 0.65, 0.59, and 0.67, I can reject the null hypothesis. So, I conclude that these correlation coefficients are not a chance finding and in fact are statistically significant. The same explanation shows that all correlation coefficients presented in Table 11 are statistically significant.

5.3.2 Significance Tests on Correlations

One of the traditional tests for significance has been *t-test*. However, Steiger [Steiger, 1980] in his work on comparing correlated correlation coefficients introduces some of the potentially serious problems with *t-test*. But apparently, even after Steiger's warning, the *t-test* is still the standard test for comparing two correlated correlation coefficients. Even Hotelling, who proposed *t-test*, in his paper [Hotelling, 1940] declares such a warning "The advantages of exactness and of freedom from the somewhat special trivariate normal assumption are attained at the expense of sacrificing the precise applicability of the results to other sets of the predictors". This means that although the *t-test* has been proved to be effective in measuring statistical significance of a difference between correlation scores calculated on **different samples**, it is not a reasonable choice for comparing the significance of a difference between correlation scores of **different predictors** [Meng *et al.*, 1992]. This is the case when three variables, x, y, and z, are given and we want to compare the r_{xy} with the r_{zy}. It comes up when you want to know which of two variables: x, and z are more related to a third variable y (the dependent variable) on the same sample data, and is called "overlapping correlations".

Therefore, in this work I use two other measures for the sake of statistical significance test. First, I will compare the *Confidence Interval* (CI) of the correlation coefficients achieved by each of the methods, using *Fisher's z transformation* [Dunn and Clark, 1969]. Second, I will go a step further

and conduct a statistical significance test on the difference between correlation coefficients achieved by *Standard Learning*, *Standard Learning with Feature Selection* and the *Two Phase Learning*. For this latter purpose I apply the method proposed by Zou [Zou, 2007] which is based on *Fisher's z transformation* with some adaptations.

Fisher

In [Dunn and Clark, 1969] a test for comparing two correlated correlation coefficients using Fisher's z transformation was proposed. In the proposed method, using *Fisher's z transformation* each correlation coefficient (shown by r) gets transformed into z scale, where z is normally distributed with the mean of:

$$z_r = \frac{1}{2} \ln \frac{1+r}{1-r}$$

and has the standard error (standard deviation) of:

$$SE = \sqrt{\frac{1}{N-3}}$$

where N is the size of the sample. The confidence interval on z scale is defined by:

$$CI_{z-scale} = z_r \pm 1.96 \times SE$$

Note that the 1.96 is because 95% of the area under a normal curve lies within roughly 1.96 standard deviations of the mean, and due to the central limit theorem, this number is therefore used in the construction of approximate 95% confidence intervals.

The calculated CI is now in z-scale which needs to be transformed back into r-scale using the

following formula:

$$CI_{r-scale} = \frac{e^{2z} - 1}{e^{2z} + 1}$$

By following this procedure, CIs were calculated and presented in Table 13. For example comparing the confidence interval for the *Standard Learning* method, presented by *Standard-CI* on *OnWN*, with the confidence interval for the *Two Phase Learning*, shown by *Two Phase-CI*, demonstrates that although there is overlap between the intervals, the *Two Phase Learning* has an interval of greater possible correlation coefficients: $(0.62, 0.70)$ V.S. $(0.68, 0.75)$. Also as it is demonstrated, the CIs of the *Standard Learning*, and *Standard Learning with Feature Selection* methods on 4 out of 5 test sets are almost the same as the confidence intervals of the 2012 best system *UKP-run2*.

	MSR-Paraphrase	MSR-Video	SMTeuroparl	SMTnews	OnWN
Test Set's Size - *N*	750	750	459	399	750
Standard Correl.	0.65	0.87	0.51	0.36	0.66
Standard CI (r scale)	(0.60,0.69)	(0.85,0.88)	(0.44,0.57)	(0.26,0.44)	(0.62,0.70)
Feature Selection Correl.	0.59	0.87	0.52	0.40	0.67
Feature Selection CI (r scale)	(0.53,0.62)	(0.85,0.88)	(0.45,0.58)	(0.30,0.47)	(0.62,0.70)
Two Phase Correl.	0.67	0.89	0.56	0.43	0.72
Two Phase CI (r scale)	(0.63,0.71)	(0.87,0.90)	(0.49,0.62)	(0.34,0.50)	(0.68,0.75)
UKP-run2 (best) Correl.	0.68	0.87	0.52	0.49	0.66
UKP-run2 (best) CI (r scale)	(0.63,0.71)	(0.85,0.88)	(0.45,0.58)	(0.40,0.59)	(0.62,0.70)
STS-2012 baseline Correl.	0.43	0.29	0.45	0.39	0.58
STS-2012 baseline CI (r scale)	(0.36,0.48)	(0.23,0.35)	(0.37,0.51)	(0.30,0.47)	(0.53,0.62)

Table 13: Confidence Intervals for correlation coefficients presented in Table 11

As it was shown, conducting this test only requires the correlation coefficients and the size of the datasets. Therefore I could calculate the CI for each system's correlation coefficients. The same method was used in the STS shared task-2012 [Agirre *et al.*, 2012] as the statistical significance test to compare systems' output. They report that according to the CIs of the first and the second best systems of 2012 shared task, the first system did not have any statistical advantage over the second system. As can be seen, according to these CIs, the *Two Phase Learning* in 2 of the test sets (MSR-Paraphrase, and MSR-Video) performs equal to the *UKP-run2* system, in 1 of the surprise test sets (SMTnews) performs worse, and in the remaining 2 (SMTeuroparl, OnWN) statistically outperfroms the *UKP-run2* system.

Zou

Confidence intervals are widely accepted as a preferred way to present study results. They encompass significance tests and provide an estimate of the magnitude of the effect. However, comparisons of correlations still relies on significance testing.

As I explained earlier, the confidence interval for a single correlation r is often obtained using Fisher's r to z transformation. Zou [Zou, 2007] explains that using the same idea for measuring the difference between two correlation $(r_1 - r_2)$ will fail because the limits for $z_{r_1} - z_{r_2}$ cannot be back-transformed to obtain the interval for $r_1 - r_2$.

Zou presents a general approach for constructing confidence intervals for a difference between correlations. The proposed method presents a general approach to constructing approximate confidence intervals for differences between 2 overlapping correlations. This approach requires the availability of confidence limits for the separate correlations and, for correlated correlations, a method for taking into account the dependency between correlations. Since calculating correlation between correlations requires the access to each systems' output, I could only conduct this test on *Standard Learning*, *Standard Learning with Feature Selection* and our proposed method.

Zou defines the correlation between two correlations r_1 and r_2 as:

$$\widehat{corr}(r_1, r_2) = \frac{(r_{12} - 0.5 r_1 r_2)(1 - r_1^2 - r_2^2 - r_{12}^2) + r_{12}^3}{(1 - r_1^2)(1 - r_2^2)}$$

Then using the CIs calculated by Fisher's transformation, the lower bound of the confidence interval for $r_1 - r_2$ is defined by:

$$L = r_1 - r_2 - \sqrt{(r_1 - l_1)^2 + (u_2 - r_2)^2 - 2\widehat{corr}(r_1, r_2)(r_1 - l_1)(u_2 - r_2)}$$

and the upper bound is defined by:

$$U = r_1 - r_2 + \sqrt{(u_1 - r_1)^2 + (r_2 - l_2)^2 - 2\widehat{corr}(r_1, r_2)(u_1 - r_1)(r_2 - l_2)}$$

By plugging the numbers presented in Table 13 and the $\widehat{corr}(S,T)$, $\widehat{corr}(FS,T)$ rows of the Table 14 into the above formulas, CIs of the differences between *Standard Learning* (S) and *Two Phase Learning* (T), and *Standard Learning with Feature Selection* (FS) and *Two Phase Learning* (T) on each test set are calculated and presented in Table 14. Note that these CIs are calculated for $r_{Standard} - r_{TwoPhase}$, and $r_{StandardwithFeatureSelection} - r_{TwoPhase}$.

	MSR-Paraphrase	MSR-Video	SMTeuroparl	SMTnews	OnWN
S Correl.	0.65	0.87	0.51	0.36	0.66
S CI (r scale)	(0.60,0.69)	(0.85,0.88)	(0.44,0.57)	(0.26,0.44)	(0.62,0.70)
FS Correl.	0.59	0.87	0.52	0.40	0.67
FS CI (r scale)	(0.53,0.62)	(0.85,0.88)	(0.45,0.58)	(0.30,0.47)	(0.62,0.70)
T Correl.	0.67	0.89	0.56	0.43	0.72
T CI (r scale)	(0.63,0.71)	(0.87,0.90)	(0.49,0.62)	(0.34,0.50)	(0.68,0.75)
Correl. (T&S)	0.025	0.04	0.006	-0.064	0.037
$\widehat{corr}(S,T)$	-0.077	3.78	-0.11	-0.06	-0.009
$(r_S - r_T)$ CI	(-0.09,0.04)	(-0.052,0.012)	(-0.15,0.05)	(-0.19,0.05)	**(-0.11,-0.003)**
Correl. (T&FS)	0.034	0.024	-0.0004	-0.092	0.057
$\widehat{corr}(FS,T)$	-0.091	3.94	-0.12	-0.169	-0.019
$(r_{FS} - r_T)$ CI	**(-0.155,-0.028)**	(-0.052,0.012)	(-0.137,0.057)	(-0.161,0.093)	(-0.109,0.0004)

Table 14: Confidence Intervals of differences between Standard Learning (S) and Two Phase Learning (T) correlation coefficients $(r_S - r_T)$, and Standard Learning with Feature Selection (FS) and Two Phase Learning (T) correlation coefficients $(r_{FS} - r_T)$.

These results indicate that *Two Phase Learning*, compared to *Standard Learning*, may be more predictive although the difference in correlations in 4 of the test sets, MSR-Paraphrase, MSR-Video, SMTeuroparl, and SMTnews, did not reach statistical significance, because the confidence intervals contain 0. Which means that on 4 of the test sets the performance of these two methods are close to each other. However, on one of the test sets, OnWN test set, the confidence intervals reach statistical significance. Also, as it can be seen on the test set MSR-Paraphrase, *Two Phase Learning* statistically outperforms the *Standard Learning with Feature Selection* method. However, in the remaining 4 test sets, the results achieved by our method are not statistically better than those of *Standard Learning with Feature Selection*.

5.4 STS-2013 Experiments

Our submissions for STS-2013 contained two main types of setups: *Standard Learning* (RUN-1), and *Two Phase Learning* (RUN-2, RUN-3). For the *Standard Learning* setup, one regressor was trained on the training set with all 11 features and tested on the test sets. While for the remaining runs the *Two Phase Learning* method was used. In all the submissions I use the same *Basic Features*. Also, the only difference between the *RUN-2*, and *RUN-3* is that in the the last submission, I wanted to examine if I could reduce the training time by reducing the number of support vectors and allowing larger training errors. This was intuitively done by decreasing the value of γ (in *RBF* kernel) from 0.01 to 0.0001, and decreasing the value of C (error weight) from 1 to 0.01. Having set these parameters for *RUN-3*, it resulted in smoother and simpler decision surface but negatively affected the performance as shown in Table 15.

5.4.1 STS-2013 Results

The results of our experiments are presented in Table 15.

	headlines	OnWN	FNWN	SMT
RUN-1	0.6774	0.7667	0.3793	0.3068
RUN-2	0.6921	0.7367	0.3793	0.3375
RUN-3	0.5276	0.6495	0.4158	0.3082

Table 15: Results of CLaC submissions in STS-2013 shared task

The results indicate that the proposed method, *RUN-2*, was successful in improving the results achieved by the *RUN-1* ever so slightly (the confidence invervals at 5% differ to .016 at the upper end) and far exceeds the reduced computation version of RUN-3.

5.4.2 Difference between STS 2013 and STS 2012 Experiments

In 2013 submissions I combined all training sets and formed one single training set. This is the training set that I used in all of our submissions for the 2013 task. However, in the experiment on 2012 datasets, for any test set that I were given a training set drawn from the same source, I only

used that particular training set. For the remaining test sets (surprise sets), I used the combination of all training sets as the training data.

5.5 Discussion

We compared our proposed method with the best performing system of the STS-2012 shared task, *UKP-run2* [Bär *et al.*, 2012], and showed that by going beyond a single-layer standard learning approach we could achieve improvements. Our proposed method outperformed this best system on average, while according to Fisher's CIs, on 2 of the test sets the improvements that we achieved are statistically significant.

Besides, on 2012 datasets, we conducted two other experiments: *Standard Learning* and *Standard Learning with Feature Selection*, and compared our proposed method with other possible solutions/methods for training a model. The results of Zou significance test indicated that the proposed method statistically outperforms the *Standard Learning* on OnWN, and the *Standard Learning with Feature Selection* on MSR-Paraphrase test sets. In fact, the proposed method may be still more predictive on the remaining test sets. Because, the ratio of the magnitude of the positive side of the CIs to the magnitude of the negative side is very low (For example, for the CI $(-0.109, 0.0004)$ this ratio is 0.003). Since we calculated the intervals based on $(r_S - r_T)$ and $(r_{FS} - r_T)$, the positive side of the interval corresponds to the situation that the other methods, could beat our proposed method. We also emphasized the importance of using a proper statistical significance test and going beyond the blind usage of well-known tests, such as *t-test*, by understanding the differences between the nature of tasks.

Finally, we participated in STS-2013 shared task and ranked 4th among 34 teams. This final result and our results from STS-2012 experiments indicates that the proposed system achieves the level of performance that is comparable to the state-of-the-art systems. In the next chapter, we will discuss some of the outcomes of the 2013 shared task along with the error analysis of the results from 2012 experiments.

Chapter 6

Analysis

In this section I want to go a step further and talk briefly about the data and some the outcomes of my experiments. In this regard, I will first talk about error cases that this feature space and method failed to predict correctly. The 2012 experiments are used for this purpose. Then in the end of this chapter, I will briefly talk about some of the findings of the STS-2013 experiments.

6.1 Error Analysis

It is important to mention that the purpose of the error analysis section is not to explain why the system failed on some cases, but it is to recognize which aspects of the language, frequently occurring in these datasets, were difficult to address using a lexical-oriented system. So, the section is about identifying those aspects and not about the reason of incorrect prediction. Nonetheless, I believe that by identifying those linguistic phenomena it would be clear why the proposed feature space failed in some cases.

Therefore, I need to define/answer two things: First and foremost, "what should be considered as an error?". Second, "how many times a pair should be caught as an error, to be considered as a frequent error case?". I will try to answer these questions in Sections 6.1.1, and 6.1.2.

I use the training data for the error analysis, because the purpose of error analysis is to actually

characterize the error cases without concern for the limitations of machine learning methods, which is the possibility of failure in dealing with unforeseen data.

In other words, if an error occurs in the surprise sets, two of the test sets, it can be because of the lack of the training set for that particular type of data, or the failure of the feature space in addressing that instance. Since it is impossible to distinguish the reason of the failure, by focusing on the training sets I could eliminate the first possibility. Therefore, I trained 11 SVRs using each of the 11 *Basic Features*, 3 SVRs using the 3 combined features (string*, ROUGE*, kb*), and 1 SVRs using all 11 *Basic Features* together.

6.1.1 Definition of Error

An error is defined as a difference between the desired and the actual output. If I only deal with categorical outputs, like discrete numeric values or class labels, catching error cases is a clear task. Also, if I only care about the amount of error that the system made, I can calculate and sum the difference between all actual values and their corresponding predicted values.

However, in this work, I deal with continuous predictions and therefore, the task of catching error cases becomes a troublesome task. Given that in a continuous space it is not possible for the system to output exactly the same value as the desired value, if I make the judgment based on this naive approach that any value that does not exactly match the desired value is an error I will catch almost all the instances as errors.

In order to overcome this issue, I propose an approach for catching errors which originates from the instruction that was provided by the task organizers. Have in mind that the output interval for the STS task should be [0-5]. Before going any further, let's study what they haves proposed as the basis for their grading schema:

score: 5 The two sentences are completely equivalent, as they mean the same thing.

The bird is bathing in the sink.

Birdie is washing itself in the water basin.

score: 4 The two sentences are mostly equivalent, but some unimportant details differ.

In May 2010, the troops attempted to invade Kabul.

The US army invaded Kabul on May 7th last year, 2010.

score: 3 The two sentences are roughly equivalent, but some important information differs/missing.

John said he is considered a witness but not a suspect.

"He is not a suspect anymore." John said.

score: 2 The two sentences are not equivalent, but share some details.

They flew out of the nest in groups.

They flew into the nest together.

score: 1 The two sentences are not equivalent, but are on the same topic.

The woman is playing the violin.

The young lady enjoys listening to the guitar.

score: 0 The two sentences are on different topics.

John went horse back riding at dawn with a whole group of friends.

Sunrise at dawn is a magnificent view to take in if you wake up early enough for it.

If I look at the example for score 4, I realize that the temporal information *May 7th*, and the named entity *US* are considered as unimportant details. Let's assume that this is a correct assumption and try to judge between the pair of sentences in below:

(6.1) (a) In 2011, people voted NDP.

(b) In 2009, people voted NDP.

If I only take the main event *Vote*, the agent *people*, and the theme *NDP* as important information I will consider both sentences as mostly equivalent and the corresponding score would be 4. However one can recognize that even though the main event, the agent, and the theme of both sentences are the same, their times of occurrence are completely different, which distinguishes them as two different events. In this regard, I can see that even making decision about the importance of one aspect of language is a very difficult and deep task.

Going back to the starting point of this small argument, where I wanted to propose an approach for identifying errors. I decided to use a sort of abstract interpretation on top of the grading schema

that was provided by the task organizers. This abstraction splits the grading schema into two separate parts. One part is representative for pairs that are identified as *relatively equivalent* and covers the pairs with the the scores in range of **[3-5]**. The other part is representative for pairs that are *not equivalent* and covers those that lie in the range of **[0-2]**. I left the range of [2-3] as a gray area that I can not say that much about, in the error analysis step.

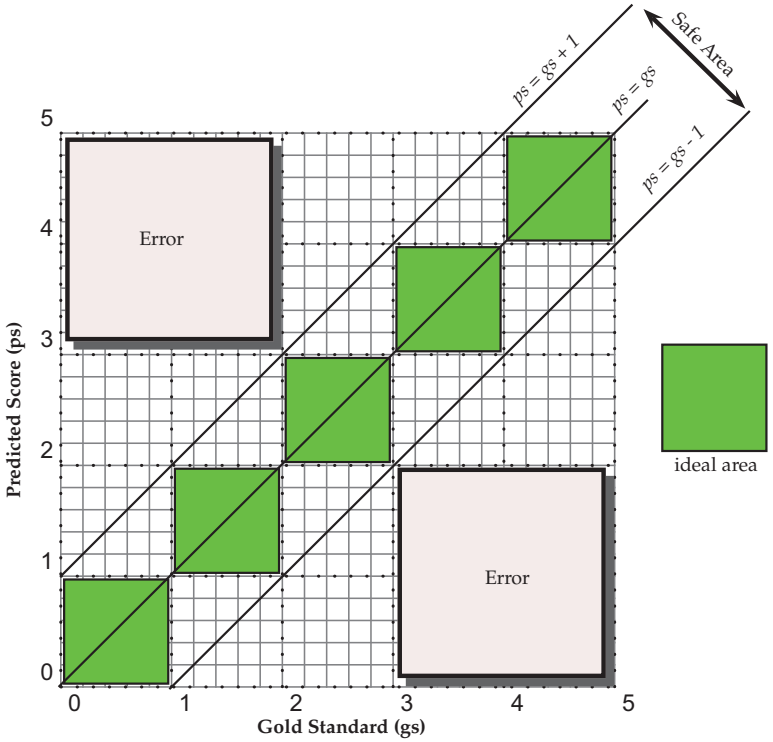


Figure 5: A mapping from the grading schema to the proposed error schema.

According to this approach I define an error case to be the case that was expected to be mapped to one part, according to the gold standard, but the systems predicted as a member of the other part. A visualization of the proposed error schema is provided in Figure 5.

In Figure 5, the top left "Error" square represents cases for which the gold standard score is in the range of [0-2] while the predicted score is in the range of [3-5]. The bottom right "Error" square represents those cases where the gold standard score is in the range of [3-5] while the system's output is in the range of [0-2]. The boxes which are lined up along the bisector line *ps=gs* are ideal areas

were the predicted score and the gold standard scores are very close to each other. The so-called "Safe Area" is the area where the difference between the gold standard and the predicted scores are less than 1. The error cases that I am going to put under my microscope are those that are represented by "Error" boxes.

6.1.2 Error Cases

Now that I provided the definition of an error case and the error schema, I can take a look at the dataset and each experiment to see which error cases have occurred more frequently across different experiments. I also want to identify the frequently occurring linguistic phenomena that the lexical-oriented system fails to address.

Moreover, I need to set a threshold on the number of times a pair has to occur as an error during different experiments in order to be caught as a frequent error. Then I look at each error case and the features that were used in those experiments that resulted in incorrect prediction for that particular error case.

Threshold

As mentioned earlier, using the proposed error schema, I also need to specify a threshold value which corresponds to the number of times a pair was captured as an error case. The maximum number of error occurrence is 15, which equals the number of experiments that I did on each training set, while the minimum is 0. For example, if I use the threshold of 8 it means that I am considering those pairs that have been captured as an error in at least 8 out of 15 experiments. This basically means that if I set a threshold of 8, I am considering those error cases that at least occurred in half of the experiments.

I tried different thresholds just to get a reasonable accumulative number of error cases that I can manually investigate. For the thresholds of 7, 10, and 15 I obtained 288, 160, and 26 errors respectively. Since the number of errors for the threshold of 15 is a reasonable number I selected these error cases as the main resource of evaluation. It also has a nice interpretation, it means that

I am taking into consideration those cases that are always in error. In another words, these cases are the ones that our system, no matter what lexical features are being used, always fails to predict correctly.

Error Categories

In this section, I study some of these error cases and try to group them into some categories based on their common characteristics.

NUMBERS In 10 of the error cases the numbers were the main or one of the main reasons of the failure. For example in the following pair, despite the word overlaps, numbers and the predicate differentiate the meaning of the two sentences:

(6.2) pair_id:689 from MSR-Paraphrase gs:1.75

(a) The jury awarded TVT about $23 million in compensatory damages and roughly $108 million in punitive damages.

(b) TVT Records sought $360 million in punitive damages and $30 million in compensatory damages, officials said.

NAMED-ENTITY In 3 of the error cases named-entities were the main or one of the main reasons of the failed prediction. For example in the following pair, the *person's name* and *locations* distinguish the contents of the two sentences:

(6.3) pair_id:459 MSR-Paraphrase gs:1.8

(a) Russ Britt is the Los Angeles Bureau Chief for CBS.MarketWatch.com.

(b) Emily Church is London bureau chief of CBS.MarketWatch.com.

Or in the following pair the mismatched named-entities and temporal constraints separate the meaning of the two sentences:

(6.4) pair_id:237 MSR-Paraphrase gs:1.75

(a) Board Chancellor Robert Bennett declined to comment on personnel matters Tuesday.

(b) *Mr. Mills declined to comment yesterday, saying that he never discussed personnel matters.*

QUOTATION In 3 of the error cases quotations were the main reasons of the failed prediction. For example in the following pair, quotations do not match while the rest of the arguments are identical:

(6.5) pair_id:275 MSR-Paraphrase gs:1.75

(a) *"Is it in the food supply?" says David Ropeik, director of risk communication at the Harvard Center for Risk Analysis.*

(b) *"It's not zero," said David Ropeik, director of risk communication at the Harvard Center for Risk Analysis.*

The same thing applies to the this pair:

(6.6) pair_id:595 MSR-Paraphrase gs:1.8

(a) *About 100 firefighters are in the bosque today, Albuquerque Fire Chief Robert Ortega said.*

(b) *"We were seconds away from having that happen," Albuquerque Fire Chief Robert Ortega said.*

PREPOSITIONAL PHRASE In 2 of the error cases prepositional phrases were involved. For example in the following pair despite the similarity in the topic and the predicates of the two sentences the difference in the prepositional phrase causes the incorrect prediction:

(6.7) pair_id:462 MSR-Video gs:3

(a) *The man is throwing knives at a tree.*

(b) *A man is throwing blades into a close target outside.*

Also in the following pair, the presence of the prepositional phrase in one sentence and its absence in the other sentence, while two sentences are very close in meaning, effects the similarity of the sentences and results in failed prediction:

(6.8) pair_id:733 MSR-Video gs:3.5

(a) *A monkey is karate kicking at someone's gloved hand.*

(b) *A monkey practices martial arts.*

6.1.3 Discussion

The error cases are good examples to show how easy it could be to fool a lexical-oriented system despite its robustness in providing a reliable and consistent performance on large-scale data. In fact, while two sentences are almost identical in terms of their lexical units, a slight difference in *numbers*, *temporal constraints*, *quotations' content*, etc, can change the meaning of a text.

Despite this disadvantage, improving the results achieved by a lexical-oriented system by incorporating deeper semantic features is a very difficult task. Because, in addition to annotating these deeper phenomena, one also needs to find a way to compare them. For example considering two phrases "2034 pairs" and "more than 2000 pairs", finding a reliable interpretation/representation that catches these two as similar is a very difficult task. For example, assume the simple heuristics which replaces "more than 2000" with "$X > 2000$". It can successfully identify the similarity between these two phrases. Now assume that we are given the third phrase "1500000 pairs", the previous heuristics incorrectly identifies the second and third phrases as equivalent while they are semantically very different.

Also, finding a way to incorporate these phenomena into a statistical/machine learning method is very challenging. Although having a system that annotates the aforementioned phenomena has many merits in some specific tasks (engineered for a particular purpose), in general domains these phenomena do not occur as often as simple lexical units. Therefore, even by using a very comprehensive approach for annotating and comparing them, we may not achieve any noticeable improvement or due to their sparsity, may even see decrease in the performance previously achieved by a purely lexical-based system.

6.2 Analysis of STS-2013

Since I have trained separate models based on each subset of features, for further analysis I could use the predicted scores generated by each of these models and calculate their correlations. This way I could see that actually which of the feature combinations were more effective in making prediction on which test set. In Table 16, I list the best and worst feature combination that gives the highest and the lowest performance on each test set.

	headlines	
	Best	Worst
Combination	[esa lem ROUGE-1 ROUGE-SU4]	[jcn lem lcsq]
Correlation	0.7329	0.3375
	OnWN	
	Best	Worst
Combination	[esa lin jcn roget lem lcsq ROUGE-1 ROUGE-W ROUGE-SU4]	[jaro]
Correlation	0.7768	0.1425
	FNWN	
	Best	Worst
Combination	[roget ROUGE-1 ROUGE-SU4]	[ROUGE-2 lem lcsq]
Correlation	0.4464	-0.0386
	SMT	
	Best	Worst
Combination	[lin jcn roget ROUGE-1]	[esa lcsq]
Correlation	0.3648	0.2305

Table 16: Best and Worst combination for each test set

As you may notice, these results indicate that *ROUGE-1* (denoted by RO-1), which is the feature based on unigram, in all of the 4 best performing subsets exists. Also, the features *ROUGE-SU4*, and *Roget's* in three of the best subsets exist, while *esa*, *lin*, *jcn* are part of the two of the best subsets. Also, the "worst" column indicates that *lcsq* was not an effective feature.

In Table 17 I list all the features and, instead of looking at the best combination, take the top three best combinations for each test and see how many times each feature has occurred in top 12 combinations (first column). Further I divide the test sets into short (OnWN,headlines) and long (FNWN,SMT), and see features behavior in each of these two categories (denoted by short and long). The last column reports the number of time a feature has occurred in the best combinations (out of 4).

Features	out of 12	out of 6(short)	out of 6(long)	out of 4
esa	6	6	0	2
lin	6	3	3	2
jcn	4	1	3	2
roget	9	3	6	3
lem	6	6	0	2
jaro	0	0	0	0
lcsq	3	3	0	1
ROUGE-W	7	4	3	1
ROUGE-1	10	6	4	4
ROUGE-2	3	1	2	0
ROUGE-SU4	10	5	5	3

Table 17: Feature combination analysis

A quick observation, shows that *ROUGE-1*,*ROUGE-SU4*, and *roget* are in fact effective features across different test sets. Also it demonstrates that the feature *esa*, *lem* are very reliable when we deal with short text fragments, while they do appear in the best combinations for longer text fragments. Further, features *lin*, *jcn*, and *ROUGE-W* show on-and-off behavior on large and short text fragments. Therefore, if we are only limited to pick 5 features from these lexical features, *ROUGE-1*, *ROUGE-SU4*, *roget* should be definitely tried. And if we know that the data contains only short text fragments, *esa*, *lem* are safe choices, while if we have no intuition about the type of the data, *lin*, *jcn*, *ROUGE-W* are good candidates to provide reasonable performance.

Chapter 7

Conclusion

Sentence similarity is one of the core elements of Natural Language Processing (NLP) tasks such as Recognizing Textual Entailment, and Paraphrase Recognition. Over the years, different systems have been proposed to measure similarity between fragments of texts. In this research, we propose a new two phase supervised learning method which uses a combination of lexical features to train a model for predicting similarity between sentences. Each of these features, covers an aspect of the text on implicit or explicit level. The two phase method uses all combinations of the features in the feature space and trains separate models based on each combination. Then it creates a meta-feature space and trains a final model based on that. The proposed method contrasts existing approaches that use feature selection, in the sense that it does not aim to find the best subset of the possible features. In fact, the *Phase Two Model* of our proposed approach, learns the relationship between each *Phase One Model's* prediction and the gold standard, instead of learning the relation between one single possibility of the features and the gold standard values. This is beneficial because a *Phase One Model* might consistently make the correct predictions for a particular class, while another model may perform poorly on that class.

I compared our proposed method with state-of-the-art systems and showed that by going beyond a single-layer standard learning approach we could achieve improvements. I also discussed some of

the issues with a lexical-oriented system in Chapter 6. I showed how easy it is to fool a lexical-oriented system despite its robustness in providing a reliable and consistent performance on large-scale data. In fact, while two sentences are almost identical in terms of their lexical units a slight difference in *numbers*, *temporal constraints*, *quotations' content*, etc, can considerably shift the meaning of a text. Despite this disadvantage, improving the results achieved by a lexical-oriented system by incorporating deeper semantic features is a very difficult task. Because, in addition to identifying these deeper phenomena, one also needs to find a way to compare them. Also, finding a way to incorporate these phenomena into a statistical/machine learning method is very challenging. In fact, although having a system that annotates the aforementioned phenomena has many merits in some specific tasks (engineered for a particular purpose), in general domains these phenomena do not occur as often as simple lexical units. Therefore, even by using a very comprehensive approach to handle them, we may not achieve any noticeable improvement or may even see decrease in the performance previously achieved by a purely lexical-based system.

Furthermore, we showed that the two-phase exhaustive model, while resource intensive, is not at all prohibitive. If the knowledge to pick appropriate features is not available and if not enough training data exists to perform feature selection, the exhaustive method can produce results that are comparable to the state-of-the-art systems. But more importantly, this method allows us to forensically analyze feature combination behavior. I was able to establish that unigrams and 4-skip bigrams are most versatile features, but surprisingly that Rogets Thesaurus outperforms the two leading WordNet-based distance measures. In addition, ROUGE-W, a weighted longest common subsequence algorithm that to our knowledge has not previously been used for similarity measurements shows to be a fairly reliable measure for all data sets, in contrast to longest common subsequence, which is among the lowest performers. I feel that the insight I gained well justified the expense of our approach.

The proposed method is presented here as an interesting methodology for exploration. In fact, the current method and the developed pipeline, gives us the possibility of examining the effect of

incorporating deeper semantics into a lexical-oriented system. Also, in order to test the robustness and reliability of the proposed method, using the *Two Phase Learning* for different similar tasks such as paraphrase recognition are possible extensions to this work.

Bibliography

[AbdelRahman and Blake, 2012] S. AbdelRahman and C. Blake. Sbdlrhmn: A rule-based human interpretation system for semantic textual similarity task. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*, 2012.

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

[Allison and Dix, 1986] L. Allison and T.I. Dix. A bit-string longest-common-subsequence algorithm. *Information Processing Letters*, 23(5), 1986.

[Baeza-Yates *et al.*, 1999] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 82. Addison-Wesley New York, 1999.

[Banea *et al.*, 2012] C. Banea, S. Hassan, M. Mohler, and R. Mihalcea. UNT: A supervised synergistic approach to semantic text similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*, 2012.

[Banerjee and Pedersen, 2003] S. Banerjee and T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *International Joint Conference on Artificial Intelligence*, volume 18. Lawrence Erlbaum Associates LTD, 2003.

[Bär *et al.*, 2012] D. Bär, C. Biemann, I. Gurevych, and T. Zesch. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*, 2012.

[Bendersky and Croft, 2009] M. Bendersky and W.B. Croft. Finding text reuse on the web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. ACM, 2009.

[Broder, 1997] A.Z. Broder. On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences*, 1997.

[Budanitsky and Hirst, 2006] A. Budanitsky and G. Hirst. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1), 2006.

[Callison-Burch *et al.*, 2008] Chris Callison-Burch, Philipp Koehn, Christof Monz, Josh Schroeder, and Cameron Shaw Fordyce, editors. *Proceedings of the Third Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Columbus, Ohio, June 2008.

[Clough *et al.*, 2002] P. Clough, R. Gaizauskas, S.S.L. Piao, and Y. Wilks. Meter: Measuring text reuse. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2002.

[Cohen *et al.*, 2003] W.W. Cohen, P. Ravikumar, S.E. Fienberg, et al. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Information Integration on the Web (IIWeb-03)*, 2003.

[Cunningham *et al.*, 2011] Hamish Cunningham, Diana Maynard, and Kalina Bontcheva. *Text processing with GATE*. Gateway Press CA, 2011.

[Dagan *et al.*, 2006] I. Dagan, O. Glickman, and B. Magnini. The Pascal recognising textual entailment challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, 2006.

[De Marneffe *et al.*, 2006] Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, 2006.

[Doddington, 2002] G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 2002.

[Dolan *et al.*, 2004] B. Dolan, C. Quirk, and C. Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2004.

[Dunn and Clark, 1969] Olive Jean Dunn and Virginia Clark. Correlation coefficients measured on the same individuals. *Journal of the American Statistical Association*, 64(325), 1969.

[Gabrilovich and Markovitch, 2007] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.

[Gusfield, 1997] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge Univ Press, 1997.

[Guthrie *et al.*, 2006] D. Guthrie, B. Allison, W. Liu, L. Guthrie, and Y. Wilks. A closer look at skip-gram modelling. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, 2006.

[Guyon and Elisseeff, 2003] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3, 2003.

[Hall *et al.*, 2009] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1), 2009.

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

[Hirst and St-Onge, 1998] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 13, 1998.

[Hoffart *et al.*, 2011] J. Hoffart, F.M. Suchanek, K. Berberich, E. Lewis-Kelham, G. De Melo, and G. Weikum. Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World Wide Web*. ACM, 2011.

[Hofmann, 1999a] T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1999.

[Hofmann, 1999b] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999.

[Hotelling, 1940] Harold Hotelling. The selection of variates for use in prediction with some comments on the general problem of nuisance parameters. *The Annals of Mathematical Statistics*, 1940.

[Hovy *et al.*, 2006] Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. Association for Computational Linguistics, 2006.

[Inkpen *et al.*, 2006] D. Inkpen, D. Kipp, and V. Nastase. Machine learning experiments for textual entailment. In *Proceedings of the second Recognizing Textual Entailment Challenge*, 2006.

[Inkpen, 2007] D. Inkpen. Semantic similarity knowledge and its application. *INFORMATICA*, 1, 2007.

[Islam and Inkpen, 2006] A. Islam and D. Inkpen. Second order co-occurrence pmi for determining the semantic similarity of words. In *Proceedings of the International Conference on Language Resources and Evaluation*, 2006.

[Islam and Inkpen, 2009] A. Islam and D. Inkpen. Semantic similarity of short texts. *Recent Advances in Natural Language Processing V*, 309, 2009.

[Jarmasz and Szpakowicz, 2003] M. Jarmasz and S. Szpakowicz. Rogets thesaurus and semantic similarity. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, 2003.

[Jaro, 1989] M.A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 1989.

[Jiang and Conrath, 1997] J.J. Jiang and D.W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th International Conference on Research on Computational Linguistics*, 1997.

[Jurafsky *et al.*, 2000] D. Jurafsky, J.H. Martin, A. Kehler, K. Vander Linden, and N. Ward. *Speech and language processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, volume 2. Prentice Hall New Jersey, 2000.

[Kilgarriff *et al.*, 2004] Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. Itri-04-08 the sketch engine. *Information Technology*, 105, 2004.

[Koehn *et al.*, 2007] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, 2007.

[Krestel *et al.*, 2010] Ralf Krestel, René Witte, and Sabine Bergler. Predicate-argument extractor (pax). *New Challenges For NLP Frameworks Programme*, 2010.

[Landauer *et al.*, 1998] T.K. Landauer, P.W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3), 1998.

[Leacock and Chodorow, 1998] C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2), 1998.

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

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

[Levenshtein, 1966] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8), 1966.

[Lin and Hovy, 2003] C.Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003.

[Lin and Och, 2004] C.Y. Lin and F.J. Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004.

[Lin and Pantel, 2001] D. Lin and P. Pantel. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(4), 2001.

[Lin, 1998a] D. Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, 1998.

[Lin, 1998b] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, volume 1, 1998.

[Lin, 2004] C.Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 2004.

[Lyon *et al.*, 2001] C. Lyon, J. Malcolm, and B. Dickerson. Detecting short passages of similar text in large document collections. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, 2001.

[Manning and Schütze, 1999] C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[Meng *et al.*, 1992] Xiao-Li Meng, Robert Rosenthal, and Donald B Rubin. Comparing correlated correlation coefficients. *Psychological Bulletin*, 111(1), 1992.

[Mihalcea *et al.*, 2006] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the National Conference on Artificial Intelligence*, 2006.

[Miller and Charles, 1991] George A Miller and Walter G Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1), 1991.

[Mitchell, 1997] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[Mohler and Mihalcea, 2009] M. Mohler and R. Mihalcea. 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*. Association for Computational Linguistics, 2009.

[Monge and Elkan, 1997] A. Monge and C. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proceedings of the SIGMOD Workshop on Data Mining and Knowledge Discovery*, 1997.

[Morris and Hirst, 1991] J. Morris and G. Hirst. Lexical cohesion computed by thesaural telations as an indicator of the structure of text. *Computational Linguistics*, 17(1), 1991.

[Ng, 1998] A.Y. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *Proceedings of the 15th International Conference on Machine Learning*, 1998.

[Papineni *et al.*, 2002] K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2002.

[Patwardhan, 2003] S. Patwardhan. Incorporating dictionary and corpus information into a context vector measure of semantic relatedness. Master's thesis, University of Minnesota, 2003.

[Pedersen *et al.*, 2004] T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet:: Similarity: Measuring the relatedness of concepts. In *Demonstration Papers at North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2004.

[Pulman and Sukkarieh, 2005] S.G. Pulman and J.Z. Sukkarieh. Automatic short answer marking. In *Proceedings of the second workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, 2005.

[Resnik, 1995] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995.

[Rubenstein and Goodenough, 1965] Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10), 1965.

[Salton and Buckley, 1988] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), 1988.

[Salton and McGill, 1986] G. Salton and M.J. McGill. Introduction to modern information retrieval. 1986.

[Salton *et al.*, 1975] G. Salton, A. Wong, and C.S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11), 1975.

[Salton *et al.*, 1997] G. Salton, A. Singhal, M. Mitra, and C. Buckley. Automatic text structuring and summarization. *Information Processing & Management*, 33(2), 1997.

[Šaric *et al.*, 2012] F. Šaric, G. Glavaš, M. Karan, J. Šnajder, and B.D. Bašic. TakeLab: Systems for measuring semantic text similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*, 2012.

[Schütze, 1998] H. Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1), 1998.

[Shevade *et al.*, 2000] Shirish Krishnaj Shevade, SS Keerthi, Chiranjib Bhattacharyya, and Karaturi Radha Krishna Murthy. Improvements to the smo algorithm for svm regression. *IEEE Transactions on Neural Networks*, 11(5), 2000.

[Smola and Schölkopf, 2004] A.J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3), 2004.

[Snover *et al.*, 2009] M. Snover, N. Madnani, B. Dorr, and R. Schwartz. Fluency, adequacy, or HTER? exploring different human judgments with a tunable mt metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, volume 30. Association for Computational Linguistics, 2009.

[Souza *et al.*, 2012] J. Souza, M. Negri, and Y. Mehdad. Fbk: Machine translation evaluation and word similarity metrics for semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics*, 2012.

[Steiger, 1980] James H Steiger. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2), 1980.

[Stern and Dagan, 2011] A. Stern and I. Dagan. A confidence model for syntactically-motivated entailment proofs. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, 2011.

[Turney and others, 2001] P. Turney et al. Mining the Web for Synonyms: PMI-IR Versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning*, 2001.

[Wang *et al.*, 2007] M. Wang, N.A. Smith, and T. Mitamura. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proc. of EMNLP-CoNLL*, 2007.

[Winkler, 1990] W.E. Winkler. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research Methods, American Statistical Association*, 1990.

[Wu and Palmer, 1994] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 1994.

[Zou, 2007] Guang Yong Zou. Toward using confidence intervals to compare correlations. *Psychological Methods*, 12(4), 2007.