Saarland University

Department of Computational Linguistics

Master's Thesis

Recognizing Textual Entailment

Author:
Madhumita
Matriculation: 2551615

Supervisors:
Prof. Dr. Günter Neumann
Prof. Dr. Dietrich Klakow

15th February, 2016

Declaration

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

Saarbrücken, 15th February 2016 Signature

Acknowledgements

The research for this Master's thesis was carried out at the Multilingual Technology group of the German Research Center for Artificial Intelligence (DFKI), Saarbrücken. It was partially funded by project "Excitement" of the European Commission to extend and support algorithms within the "Excitement Open Platform" (EOP). The development was done in close association with the consortium of the project.

First of all, I would like to thank my supervisor Prof. Dr. Günter Neumann for his constant support and guidance throughout. During the course of this thesis, he helped me understand the intricacies involved with research and kept me motivated. He made me reflect on my strengths and weaknesses alike. He helped and encouraged me when I got stuck, and kept me level headed when things worked well. His constant availability was a great inspiration throughout. His invaluable time, effort and experience has driven the thesis to a successful outcome.

Next, I would like to thank my supervisor Prof. Dr. Dietrich Klakow for being very supportive about the research idea and topic. I would like to thank him for providing me with the independence of working with an approach of my choice, and for analyzing my performance. I would like to further thank him for his encouragement, and for his precious time and effort during this phase. In the end, I would like to thank him reviewing the thesis proposal and thereby the final thesis, resulting in a fruitful conclusion.

Further, I would like to extend my thanks to Dr. Tae-Gil Noh for answering my numerous questions about the EOP, providing me with relevant documents for reference, and for helping me understand the technical details. I would also like to thank him for assisting me with the complexity of the Apache UIMA type systems, which are used by the Linguistic Analysis Pipelines (LAPs) of the EOP. Moreover, I would like to extend my thanks to everyone in the consortium of project "Excitement", for being patient with me whenever I needed help.

In the end, I would like to thank my family and my friends for constantly being patient and supportive. It would have never been possible without anyone of them.

Abstract

Human language is highly ambiguous and complex. It makes Natural Language Understanding (NLU) quite challenging. Knowledge about semantic inference relation between text snippets is useful for several applications. A text 'T' is said to entail a hypothesis 'H' if a human reading them can infer 'H' from 'T'.

We have developed a tool which takes (T,H) pairs as input, and returns an entailment decision as output. It is incorporated as a new Entailment Decision Algorithm (EDA) within the Excitement Open Platform (EOP). The developed EDA follows an alignment-based approach for Recognizing Textual Entailment (RTE), motivated by the third-year agenda set by the consortium of the EC-funded open source project "Excitement".

Various lexical alignment algorithms have been developed to align fragments of text and hypothesis. Approximate distance-based algorithms have been explored for token word, lemma and phrase alignment. Utilizing a distance-based match as opposed to exact match enables handling of noise in data, for example, spelling variations and noisy tokenization. Embedded word-vector based phrase alignment algorithms have additionally been developed to capture semantic information. In this algorithm, positive and negative alignments for entailment tasks are distinguished between, through the use of WordNet and VerbOcean relations. Additionally, alignment between negation words such as "not" and "none" have been identified, because they significantly affect the entailment decision.

The generated alignments are used as features for binary classification into "entailing" and "non-entailing" classes using logistic regression with L2 regularization. The EDA has been evaluated on three standard data-sets: the RTE-3 and the RTE-6 data-sets from the RTE challenges, and the recently released SNLI corpus. The RTE-3 data-set is a balanced set of 800 (T,H) pairs for training and development each. The best accuracy obtained on this data-set is 64.63% with semantic alignments. The RTE-6 data-set is a relatively larger data-set with 15,955 pairs for training, contains 95% negative cases. Approaches have been adopted to handle this imbalance in data. The best F-score obtained on the RTE-6 data-set is 42.10. Moreover, the SNLI corpus is a balanced collection of 570,000 manually annotated (T,H) pairs for 3-way entailment decisions. This data-set has been evaluated on the RTE-3 metrics, and the best obtained classification accuracy in a 2-way entailment decision task is 75.27% on the test-set.

Contents

List of Figures

List of Tables

1 Introduction

Formally known as Natural Language Understanding (NLU), making machines comprehend human languages is a research question which has gained immense popularity in recent years. It is challenging due to the complexity of natural languages. NLU is a subtopic of Natural Language Processing (NLP). It aims to understand how languages are processed in the brain, and apply the knowledge to develop intelligent systems which can emulate the process.

Several NLU tasks make use of prior information about semantic relations between multiple snippets of text to generate relevant and more accurate results. Entailment relations are few such semantic relations between pairs of text snippets. Textual entailment refers to a directional inference relationship between pairs of text expressions, denoted by 'T' (the entailing "Text") and 'H' (the entailing "Hypothesis"). 'T' is said to entail 'H' if humans reading 'T' would typically infer that 'H' is most likely true (Dagan et al., 2013).

For example, in the following pair of sentences, 'H' is entailed by 'T' because it can be inferred from 'T':

T: *As much as 200 mm of rain have been recorded in portions of British Columbia, on the west coast of Canada since Monday.*
H: *British Columbia is located in Canada.*

Similarly, the following pair holds a "non-entailment" relation between them because 'H' is not related to, and can not be deduced from 'T':

T: *Red Planet Consulting, Inc. is a full-scale project implementation firm that provides expert Smallworld software services to electric, gas and water utilities.*
H: *Mars is called "the red planet".*

These entailment relations between textual pairs are used in myriad of applications. For example, during information retrieval, query terms should be entailed by all the documents that are retrieved. Clinchant et al. (2006) have shown a significant boost in the performance of information retrieval tasks due to textual entailment. Likewise, for multi-document summarization, the sentences which are already entailed in other existing sentences in the summary can be omitted. For answering a question, the desired answer is usually found entailed in a document. Harabagiu and Hickl (2006) have shown that the use of textual entailment increases the accuracy of an open domain question answering system by as much as 20%. Moreover, very recently, Ostermann et al. (2015) have explored the usage of entailment relations for short answer scoring.

1.1 Task description and complexity

The Master's thesis aims to explore the task of textual entailment, and develop open source algorithms to decide entailment relations between pairs of text. In this section, description and complexity of the task has been discussed.

Natural language is highly variable, ambiguous and complex. Similar meanings can be expressed in multiple implicit or explicit ways. For example, the following two sentences express similar content, although they are very different at lexical level:

***T**: This incident has left me speechless. I can't be any happier at the turn of events.*
***H**: I'm elated due to the latest development.*

Similarly, the following sentence pairs are lexically and syntactically highly similar, and yet are contradictions of each other:

***T**: I am anything but a coffee addict.*
***H**: I am a coffee addict.*

The task of textual entailment is highly complex in nature because entailment decision algorithms should identify such ambiguity and variability in content, and should establish an equivalence between them.

Moreover, according to the task description for entailment, 'H' is said to be entailed in 'T', if a human being reading a (T,H) pair can infer 'H' from 'T'. This implies that in certain cases, some world knowledge apart from available content may be required to derive inference rules. For example, in the sentence pair,

***T**: Taj Mahal is located in Agra.*
***H**:Taj Mahal is located in India.*

it is assumed that the additional knowledge that Agra is located in India is available with the reader, and 'T' is said to entail 'H'.

Development of systems that address these challenges can result in increasingly intelligent systems that understand natural language, and can be of immense utility in a myriad of applications.

1.2 Alignment-based approach to recognize textual entailment

Alignment-based approaches have been extensively used in the past for machine translation, where an alignment identifies whether words are translations of each other (Brown et al., 1993). In an entailment decision task, textual alignment refers to identification

of corresponding units of text snippets between 'T' and 'H' sentences. These pairs can either hold the same meaning, or similar but different, or even contradictory meanings. The following example enlists aligned phrases in a given (T,H) pair:

***T**: John Smith rode to Seattle and bought a Honda Civic.*
***H**: John drove to Seattle.*
***Alignment pairs**: {{John Smith, John}, {rode, drove}, {to Seattle, to Seattle}}*

Identification of such alignments facilitates and supports entailment classification. Once corresponding snippets of content between 'T' and 'H' have been identified, they can be analyzed further to verify if an entailment relation holds.

An explicit alignment component within an entailment decision algorithm allows abstraction of feature identification for entailment classification, from complex algorithms that aim to understand content and identify parallel text snippets. Alignment-based approaches become increasingly popular for Recognizing Textual Entailment (RTE) in recent years.

1.3 Literature Review

In this section, the literature associated with the state-of-the-art algorithms to recognize textual entailment have been discussed. First, the RTE challenges, which were initiated to promote related research, have been described. Next, a popular open source tool for deciding entailment relations, the Excitement Open Platform (EOP) has been reviewed. Further, the systems which have been used as an inspiration and initial point for the development of the proposed system have been described. In the end, the most recent work on recognizing textual entailment, the Stanford Natural Language Inference (SNLI) system has been reviewed.

1.3.1 Recognizing Textual Entailment Challenges

Until 2005, recognition of textual entailment was performed by independent groups under application specific research. Thereby, RTE challenge was initiated as a PASCAL challenge to promote research in this field, leading towards the development of generic semantic inference engines usable across several applications.

The first RTE challenge, RTE-1 (Dagan et al., 2006), aimed to compare different approaches for textual entailment, at the same time setting benchmarks for evaluation. A balanced data-set of text-hypothesis (T,H) pairs were created for the challenge. These pairs were adopted from text snippets in the generic news domain, with respect to multiple NLP tasks - information retrieval (IR), reading comprehension, question-answering

(QA), information extraction (IE), paraphrase acquisition, machine translation. They were manually annotated as positive or negative entailment relations, and varied in terms of inference reasoning scope and difficulty. The distribution of the pairs across different tasks was not balanced. This promoted the development of generic engines. Several systems submitted their results in this challenge, with approaches ranging from logical rule based deductions to a syntactic tree distance comparison. Although the challenge was not mature yet, the response obtained encouraged several follow-up challenges.

The second PASCAL RTE challenge (Bar-Haim et al., 2006) followed on the lines of the first challenge to promote continued research in this field. The (T,H) pairs in the RTE2 data-set were compiled from actual outputs of different systems. A balanced distribution of entailing pairs was obtained from the core semantic tasks - IE, IR, QA and multi-document summarization. Results from 23 systems were submitted for an evaluation, as opposed to 17 systems earlier, which indicated a growing interest in these challenges.

The RTE-3 challenge (Giampiccolo et al., 2007) built up on the previous two RTE challenges. The challenge improved on the previous data-set to support an optional three way decision to differentiate between contradictions and unknown relations. The development set was modified to include a limited number of longer texts, forming up to a paragraph. Further, sharing a resource pool among the systems was possible. The challenge witnessed several submissions using new approaches and models, with increased accuracy. Results obtained in the RTE3 challenge encouraged moving further towards realistic distributions.

Following the popularity of the previous challenges, the next RTE challenges were tracks in the Text Analysis Conference (TAC). Extending the RTE3 task, three-way decisions were made obligatory in RTE4 challenge (Giampiccolo et al., 2009), along with the classical two-way decision task. Similarly, in the fifth RTE challenge (Bentivogli et al., 2009), the main task remained the same. However, a pilot task was introduced, which given a hypothesis, required identification of all entailing sentences in a document. Contribution of knowledge bases was additionally evaluated through ablation tests.

The sixth and seventh RTE challenge (Bentivogli et al., 2010) built up on the pilot task of the fifth RTE challenge, and introduced the difficulty of textual entailment in a more realistic corpus-oriented setup. The task aimed to explore the potential applicability of RTE engines to the problem of document summarization. The systems were required to identify of all sentences in a given corpus that entail a given hypothesis. A set of potential candidate sentences were retrieved from the corpus through Apache Lucene, which served as the candidate text sentences for this purpose. The data-set indicated the actual distribution of entailment in a corpus, where about 95% of the cases were negative.

1.3.2 Excitement Open Platform

The Excitement Open Platform (EOP) [1] (Magnini et al., 2014) is an open source software in Java language, comprising of a suite of multilingual state-of-the-art algorithms to recognize textual entailment, along with multiple linguistic resources and pipelines. The EOP was an output of project "Excitement" [2] funded by the European Commission. It implements the generic, modular and multilingual "Excitement" architecture (Padó et al., 2014) for textual entailment. Figure 1 shows this architecture.

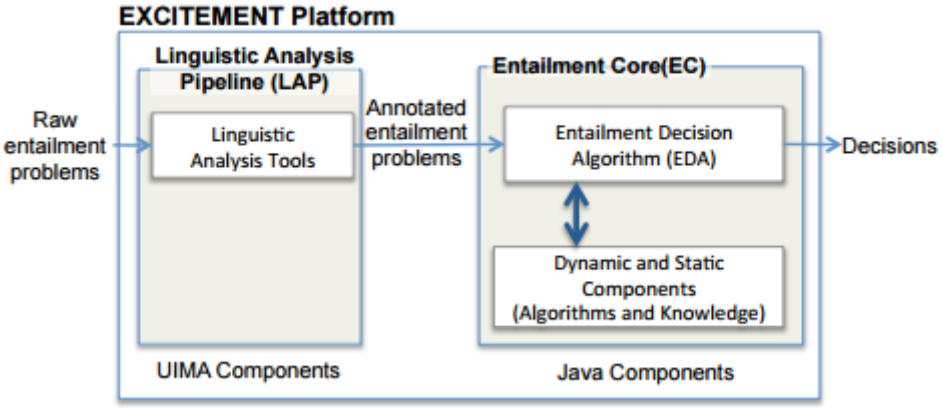


Figure 1. Excitement architecture

The "Linguistic Analysis Pipeline" (LAP) within the "Excitement" architecture abstracts annotation modules from core entailment algorithms. The LAP provides multiple pipelines which add linguistic annotations on input data. UIMA framework (Ferrucci and Lally, 2004) has been used as a container for data in DKPro (de Castilho and Gurevych, 2014) type system. The "Entailment Core" (EC) provides implementations of algorithms to recognize textual entailment. These algorithms share a common resource pool which consists of supported knowledge bases. They are based on inference rules, classification algorithms, and lexical or syntactic edit-distance between text and hypothesis.

Modular architecture of the EOP supports reuse of existing modules and development of new algorithms with minimum effort. Each "Entailment Decision Algorithm" (EDA) is an independent algorithm for deciding textual entailment. Interaction between different EDAs is not present. However, they can reuse any component implemented withing the EOP. The EOP framework specifies several interfaces to ease the development process of a complex system.

The EOP has been developed to promote experimentation with textual entailment in

[1] https://github.com/hltfbk/Excitement-Open-Platform
[2] https://sites.google.com/site/excitementproject/

varied applications. It is aimed to be utilized as a tool to generate the entailment relations within different applications, similar to tools like a part-of-speech tagger. The Moses platform (Koehn et al., 2007) used extensively in Machine Translation was an inspiration for the EOP.

In the third year of project "Excitement", multiple entailment decision algorithms within the EOP were developed through a novel explicit alignment-based classification. The next section, Section 1.3.3 describes one of these algorithms.

1.3.3 Multi-level alignments as an extensible representation basis for Textual Entailment Algorithms

This work by Noh et al. (2015) is the inspired by the third year agenda of project "Excitement". It proposes the schema in Figure 2 to abstract an alignment generation component from a component which applies these alignment links to generate an entailment decision. It extends the previous EOP architecture to include an explicit alignment component. All the alignments generated by multiple modules are unified into a single central data-structure known as "Multi-level alignments".

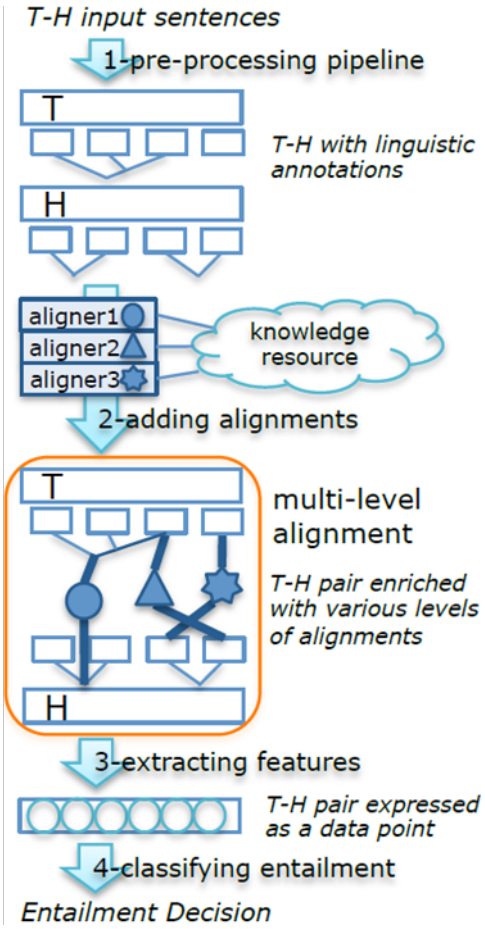


Figure 2. Data flow for entailment classification based on multi-level alignments

It is believed that different alignment links provide different information about text. These alignments may have different semantics, or may indicate different levels of alignments. Following this contention, the alignments are unified with each other, instead of selecting the preferred alignment links.

In the proposed configuration, multilingual alignments are generated. They range from semantic alignment links added via lexical resources WordNet (Miller, 1995) and VerbOcean (Chklovski and Pantel, 2004) for English, Italian WordNet (Artale et al., 1997) for Italian, and GermaNet (Hamp et al., 1997) and Geman DerivBase (Zeller et al., 2013) for German, to paraphrase alignments generated through Meteor (Denkowski and Lavie, 2011) package for machine translation. Additionally, in order to deal with Named Entities, alignment links are added between identical lemmas also.

Once the alignments have been obtained, four features have been used for entailment decision classification. These features target the coverage of words, content words, verbs and proper nouns in hypothesis by aligned content in text. These features have been developed because a higher coverage of entities in hypothesis ensure a higher probability of entailment.

Furthermore, the developed platform also supports a module to visualize the alignments produced by the system, built on the BRAT[3] library.

1.3.4 Textual Inference Engine

Textual Inference Engine (TIE) is an entailment engine developed by Wang and Neumann (2007) (Wang, 2011) at the German Research Center for Artificial Intelligence (DFKI). TIE utilizes a supervised classification approach for textual entailment. It is encapsulated within the EOP as the "Maximum Entropy Classification EDA".

In TIE, several features are generated from (T,H) pairs and multiple scores are calculated from these features. Some of these features are: bag-of-words overlap between 'T' and 'H', bag-of-lemmas overlap, an overlap between 'T' and 'H' using lexical resources WordNet (Miller, 1995) and VerbOcean (Chklovski and Pantel, 2004), dependency triple overlap and similarity between dependency trees. Thereby, a maximum entropy classifier is applied on the scores to generate an entailment decision.

1.3.5 LibLinear Textual Entailment Engine (LITE)

In the past, outside the EOP, Volokh and Neumann (2011) have used alignment scores as a feature for generating entailment decision. In their system, alignment scores have been

[3]http://brat.nlplab.org/index.html

obtained using Meteor (Denkowski and Lavie, 2011) package for Machine Translation, by assuming text and hypothesis to be translations of each other. The system relies heavily on feature templates. Several features have been identified, including an n-gram coverage, dependency triple coverage, named entity and temporal expression coverage, alignment scores and relative sentence length. Since the system assumes 'T' and 'H' to be translations of each other, relative sentence length is very important to account for alignment scores relative to entailment task. Classification decision is generated through L1 regularized logistic regression algorithm implemented in LibLinear (Fan et al., 2008). This entailment engine has been developed to target RTE-6 and 7 data.

1.3.6 Stanford Natural Language Inference

Small size of data-sets in the RTE challenges are a significant limitation for statistical systems. To overcome these limitations, Stanford Natural Language Inference corpus has been developed very recently by Bowman et al. (2015). It is a balanced data-set consisting of manually annotated 570,000 (T,H) pairs, which surpasses the existing data-sets by two orders of magnitude. It is a new benchmark for evaluation of 3-way entailment decisions, namely "entailment", "contradiction" and "neutral". This corpus supports the training of neural networks for entailment tasks, thereby promoting the applicability of the state-of-the-art research techniques to textual entailment.

The organization of the rest of the document is as follows. The proposed approach to recognize textual entailment, and the tools that will be used for this purpose are described in Section 3. In Section 4, the data-sets used, experiments conducted on the developed system, and the obtained results have been discussed. In the following section, Section 5, conclusions have been drawn from the obtained results, along with insights about improvements that can be incorporated in future. Along with that, a critical analysis of available tools and resources has been done.

2 Proposed Approach

In this section, the proposed system will be introduced. Various tools that have been used during the course of development will be discussed. Thereby, an overall architecture of the system will be put forward. Thereafter, individual components within the system, namely the alignment and the scoring components, will be elaborated upon.

Research for the proposed system has been carried out in parallel to the development of multi-level alignment architecture for textual entailment by Noh et al. (2015). It is also motivated by the third year agenda of project "Excitement" [4], which focused on development of alignment-based entailment decision algorithms. Various approaches to identify alignment links, and thereby to come to an entailment judgement, have been discussed herewith.

2.1 Overview

In this section, various approaches to identify lexical alignments will be discussed.

The most fundamental way to recognize corresponding text snippets at lexical level is an exact string match. An exact string match identifies identical pairs of words, that is, the same form of the same words that are present in both the sentences. For example, in the following sentence pair,

***T**: Müller owns a dog.*
***H**: Mueller owned a pet.*

an exact string comparison will align the 'a's in both the sentences.

Similarly, an exact root word comparison can also be used to identify pairs of corresponding words. If a sentence pair consists of the same word, used in the same way or in a different manner, this approach would add an alignment link between them. For example, in the aforementioned sentence pair, it will result in an additional alignment between "owns" and "owned".

However, more often than not, data-sets that are used are noisy. Moreover, tokenization, which refers to the process of segmenting text into pieces, is usually error prone. Furthermore, minor spelling errors and variations are frequently present in data-sets. Such variations, and errors in fundamental pre-processing steps like tokenization, significantly affect the results of systems dependent on exact-comparison based techniques. To overcome this issue, text fragment similarity can be used to identify aligned pairs of text.

[4] https://sites.google.com/site/excitementproject/

Various standard metrics to calculate similarity between two strings are the following:

- Cosine similarity: Cosine similarity between two strings x and y is calculated as,

$$cosine(x, y) = \frac{|x \cap y|}{\sqrt{|x||y|}} \tag{1}$$

- Jaccard similarity and distance: Jaccard similarity between two strings x and y is defined as,

$$J(x, y) = \frac{|x \cap y|}{|x \cup y|} \tag{2}$$

  if x and y are both empty, $J(x, y) = 1$

  Jaccard distance measures the dissimilarity between two strings. It is defined as,

$$d_J(x, y) = 1 - J(x, y)$$

- Sørensen-Dice similarity: Sørensen-Dice similarity coefficient between two strings x and y is given by the following formula,

$$s(x, y) = \frac{2|x \cap y|}{|x| + |y|} \tag{3}$$

- Overlap similarity: Overlap similarity measure calculates the overlap between two sets. For two strings x and y, it is defined as:

$$s(x, y) = \frac{|x \cap y|}{min(|x|, |y|)} \tag{4}$$

By using these metrics to calculate similarity between two strings, and by setting a similarity threshold, approximately aligned pairs of text snippets can be found. For example, in the aforementioned sentence pair, an alignment between "Müller" and "Mueller" can be generated under an appropriate similarity threshold. An exact match can be simulated through an approximate match by setting a similarity threshold of 1.0.

However, as it can be analyzed from the example, just a lexical match, whether exact or approximate, is insufficient. Deep understanding of content is required to generate meaningful semantic alignments. For example, in the previously mentioned sentence pair, the usage of words "dog" and "pet" correspond to each other and hold significant information for entailment classification.

Language specific knowledge bases can be used to identify semantic relationship between pairs of strings. These knowledge bases consist of sets of words grouped and linked

according to semantic relations between the them. However, these knowledge bases do not effectively capture the context in which various terms, words and phrases have been used. Techniques for identification of context are required to disambiguate and link these terms efficiently.

Apart from the use of linguistic resources to identify semantic relations, word and phrase vectors can also be used. Word embeddings have been introduced by Bengio et al. (2003) to learn distributed representations of words based on their context. A word embedding is a function which maps each word in a data-set to a high-dimensional vector of real numbers. These vectors are learned such that similar words have similar vector. Such a property supports the identification of semantically similar content.

2.2 Tools employed

In this section, various tools that have been used to generate the algorithms to recognize textual entailment will be discussed.

- **Excitement Open Platform (EOP)**: The EOP (Magnini et al., 2014) framework is used as an initial point for the development of new entailment decision algorithms. The developed algorithms are supported within the EOP. They reuse existing modules for linguistic pre-processing and query lexical resource database. To add linguistic annotations to the (T,H) pairs, Maltparser (Nivre et al., 2006) Lingustic Anotation Pipeline (LAP) within the EOP is queried. The annotations used during the development are token strings, token lemmas, and part of speech tags. Additionally, independent of the LAP, chunk annotations are added through the OpenNLP chunker. Various lexical resources supported by the EOP have also been explored.

- **NemexA**: NemexA is a tool developed at the Multilingual Technology lab of the German Research Center for Artificial Intelligence (DFKI). Given a multi-word entry, it finds all the similar entries in an external dictionary, based on a character n-gram vector match algorithm (Okazaki and Tsujii, 2010). It calculates similarity through a uniform distance calculation API which supports cosine, Jaccard and Dice similarity measures. Within the EDA, it is used to identify approximate distance-based alignments between pairs of token strings, lemmas and chunks of text from 'T' and 'H' respectively.

- **WordNet**: WordNet (Miller, 1995) is a large lexical database of nouns, verbs, adverbs and adjectives, grouped into sets of synonyms, linked with each other through relation between the synsets. It is used as a lexical resource to identify semantic

relations between pairs of words from T and H, specifically, to identify whether they are synonyms, hypernyms or holonyms of each other.

- **VerbOcean**: VerbOcean (Chklovski and Pantel, 2004) is a large repository of verbs, which includes the semantic relationship between those verbs. It is useful for identifying comparative strengths of verbs in T and H, which in turn aids the recognition of entailment relations.

- **Word2Vec**: Word2Vec (Mikolov et al., 2013a,b) is a tool which generates context-dependent high dimensional vectors for single and multi-word terms. It uses different architectures for this purpose, which do not involve the use of a neural net. It has been shown that the vectors obtained in such a manner capture the notions of linguistic similarity (Mikolov et al., 2013c). Similar words tend to have similar vectors. During the development of the EDA, this property is used to find semantic alignments between 'T' and 'H' phrases.

- **Weka**: Weka (Hall et al., 2009) is a collection of several machine learning algorithms and various tools for data pre-processing and visualization. These algorithms can be applied to multiple tasks related to data mining. Within the EDA, Weka is used as a machine learning library for classification of (T,H) pairs into entailing and non-entailing classes.

- **LibLinear**: LibLinear (Fan et al., 2008) is a very fast linear classifier for large datasets with millions of instances and features. It supports several SVM and logistic regression-based classification algorithms. LibLinear logistic regression algorithms for Weka have been used in the EDA for classification.

2.3 Architecture

In this section, an overall architecture of the proposed system is discussed. The system takes the input of (T,H) pairs sequentially. It first performs explicit lexical alignments between snippets of text (T) and hypothesis (H) through a distance-based match of words, lemmas and phrases. Further, it generates semantic alignments between entities through the use of knowledge bases and word embeddings. These alignments are used to calculate several scores. The calculated scores later serve as features or attributes for a supervised classifier which generates an entailment classification decision.

Complete architecture for the system has been described in Figure 3.

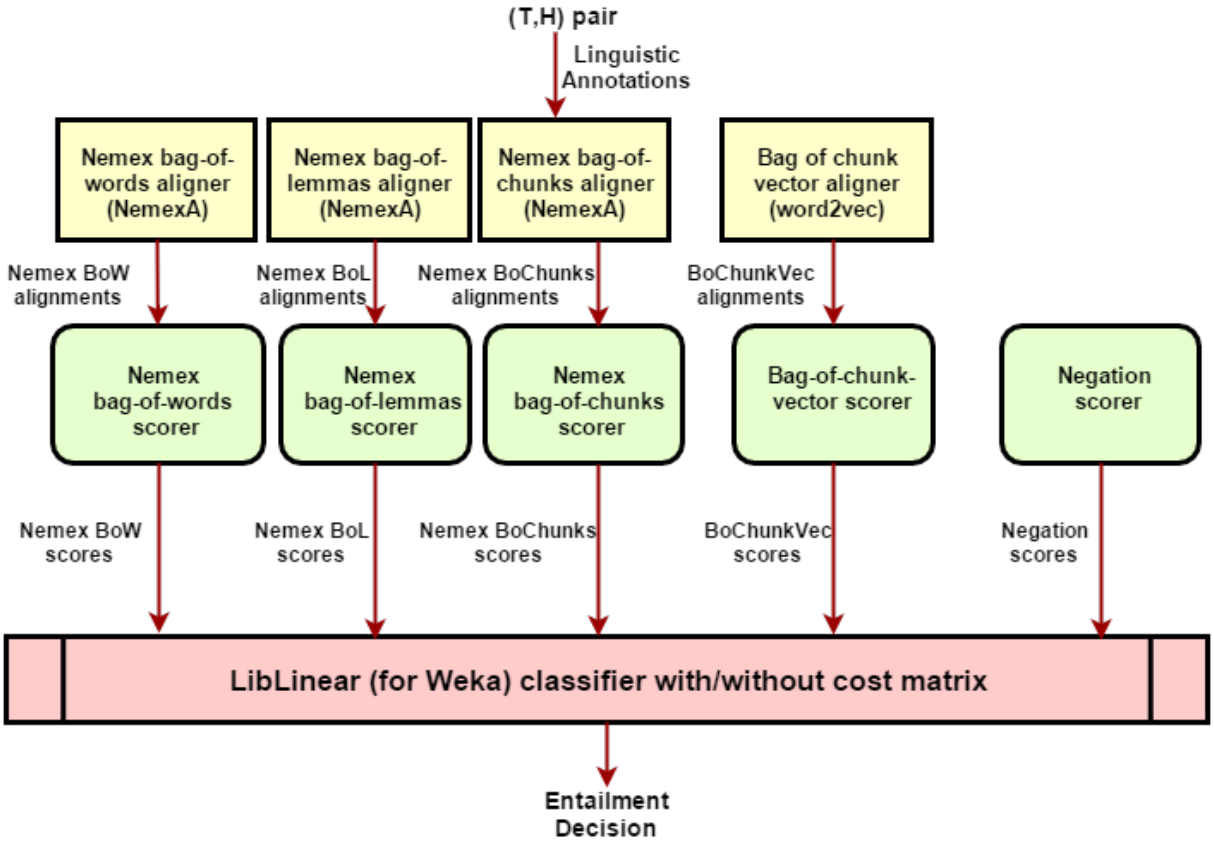


Figure 3. Nemex Classification EDA

During the training phase, all the (T,H) pairs are processed one-by-one. Scores are calculated for each pair after generating relevant alignments between segments of 'T' and 'H'. These scores are thereby written as attribute values to a separate file in the generic, native data file format supported by Weka (Hall et al., 2009), known as the ARFF file. Once all the scores for all the (T,H) pairs in the data-set are obtained, the instances present in the generated file are randomized to remove effects of the order of the (T,H) pairs on the generated model. Thereby, LibLinear (Fan et al., 2008) wrapper algorithms for Weka are used to train models for binary classification. Within LibLinear, Logistic Regression (Harrell, 2013) algorithms with L2 regularization (Ng, 2004) are used while training the classifier. Logistic regression is a machine learning technique for classifying data into discrete outcomes. It generates probabilities of outcomes, and the most probable class is assigned to the data instance. Logistic regression model finds the parameter θ to optimize the following problem:

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log p(y^{(i)}|x^{(i)};\theta) - \alpha R(\theta) \tag{5}$$

$R(\theta)$ in equation (5) is a regularization parameter. A non-zero regularization parameter typically prevents overfitting. In case of L2 regularization, $R(\theta) = \sum_{i=1}^{n} \theta_i^2$.

Instead of training a base classifier like the logistic regression classifier, cost sensitive classifiers can be trained using a cost matrix. These classifiers make their base classifier cost-sensitive. A cost matrix enables either re-weighting the training instances according to the total costs assigned to each class, or predicting the class with minimum expected misclassification cost (rather than the most likely class). This type of classification is particularly useful for training models on imbalanced data-sets. In the cases where a cost matrix is used, an optimal matrix is chosen through the use of 10-fold cross validation approach. In k-fold cross validation, the data-set is divided into 'k' subsets. Evaluations are performed by averaging the results obtained in k iterations. In each iteration, *(k-1)* subsets are used for training, and one subset is used for evaluating the trained model. Each subset is used for evaluation once.

During the testing phase, each (T,H) pair in the data-set is preprocessed sequentially yet again, to generate alignments and calculate feature scores. The same configuration is used as in the corresponding training phase. Pretrained model files are input to the same classifier, and an output classification decision is calculated and recorded individually, independent of other pairs in the testing data-set.

2.4 Nemex Aligners

Within the EDA that has been developed, approximate distance-based alignments between snippets of (T,H) pairs are identified using Nemex aligners. These aligners employ the NemexA tool for their working.

The NemexA tool is used to identify all entries from a large collection of entries, whose similarity to a given multiword entry is greater than some given threshold. It is based on the CPMerge algorithm (Okazaki and Tsujii, 2010) described below.

Formally, the algorithm finds $Y_{x,\alpha} \subseteq V$ according to the relation,

$$Y_{x,\alpha} = \{y \in V \mid sim(x, y) > \alpha\} \tag{6}$$

where V is a large vocabulary of entries, α is a given threshold, x is the query string entry, and $sim(x, y)$ is the similarity between strings x and y.

To find such entries from a very large collection, instead of finding similarity between all pairs of entries, NemexA uses a "τ-overlap join" algorithm. This algorithm reduces the size of the set of probable entries significantly, thereby making the entire process a lot faster.

First, vectors are created for each entry, using a specified n-gram size. For example, given an n-gram size 3, the vector for the term "red color" is the following:

Term: "##red#color##"
Vector: { ##r, #re, red, ed#, #co, col, or#, r## }

where '#' is used as a delimiter between the words, and '##' are used as starting and ending symbol for the given term. An inverted index is maintained to store the strings that generate a given n-gram. Now, as mentioned earlier in equation (1), a cosine distance between two n-gram feature vectors X and Y of strings x and y respectively is defined as,

$$cosine(X, Y) = \frac{|X \cap Y|}{\sqrt{|X||Y|}} \tag{7}$$

By combining equations (6) and (7), the following conditions to obtain a smaller subset of strings from the vocabulary V are derived,

$$\lceil \alpha \sqrt{|X||Y|} \rceil \leq |X \cap Y| \leq \min\{|X|, |Y|\} \tag{8}$$

$$\lceil \alpha^2 |X| \rceil \leq |Y| \leq \lfloor \frac{|X|}{\alpha^2} \rfloor \tag{9}$$

where $|X|$ and $|Y|$ refer to the size of vectors X and Y respectively. Similar steps can be followed to obtain corresponding conditions for other distance metrics also. Further, the following property of signature based algorithms (Arasu et al., 2006; Chaudhuri et al., 2006) are used to reduce the number of candidate strings even more:

Given a set X of size n and a set Y of any size. Let there be any subset $Z \subseteq X$ of size $(n - \tau + 1)$. If $|X \cap Y| \geq \tau$, then $Z \cap Y \neq \phi$.

Finally, the set of strings that satisfies all the above conditions is returned to the user. Due to large reduction in the search space, this algorithm has been found to be very fast for computing matching entries from a large database.

Now, within the Nemex aligners, the data-set consisting of (T,H) pairs are first annotated using the Linguistic Annotation Pipeline (LAP) within the EOP (Magnini et al., 2014). Depending on the type of alignment to be performed, token strings, lemmas, part-of-speech tags or chunk annotations are required. OpenNLP (Baldridge, 2005) LAP is used when lemma annotations are not required for alignment. However, in order to use the lemma annotations, the OpenNLP pipeline is insufficient and MaltParser (Nivre et al., 2006) annotation pipeline is used instead. The MaltParser pipeline uses TreeTagger (Schmid, 1994) to compute the lemma annotations. If required, chunk annotations are added using the chunker from the OpenNLP project.

The pipeline to perform an alignment through NemexA is described in Figure 4.

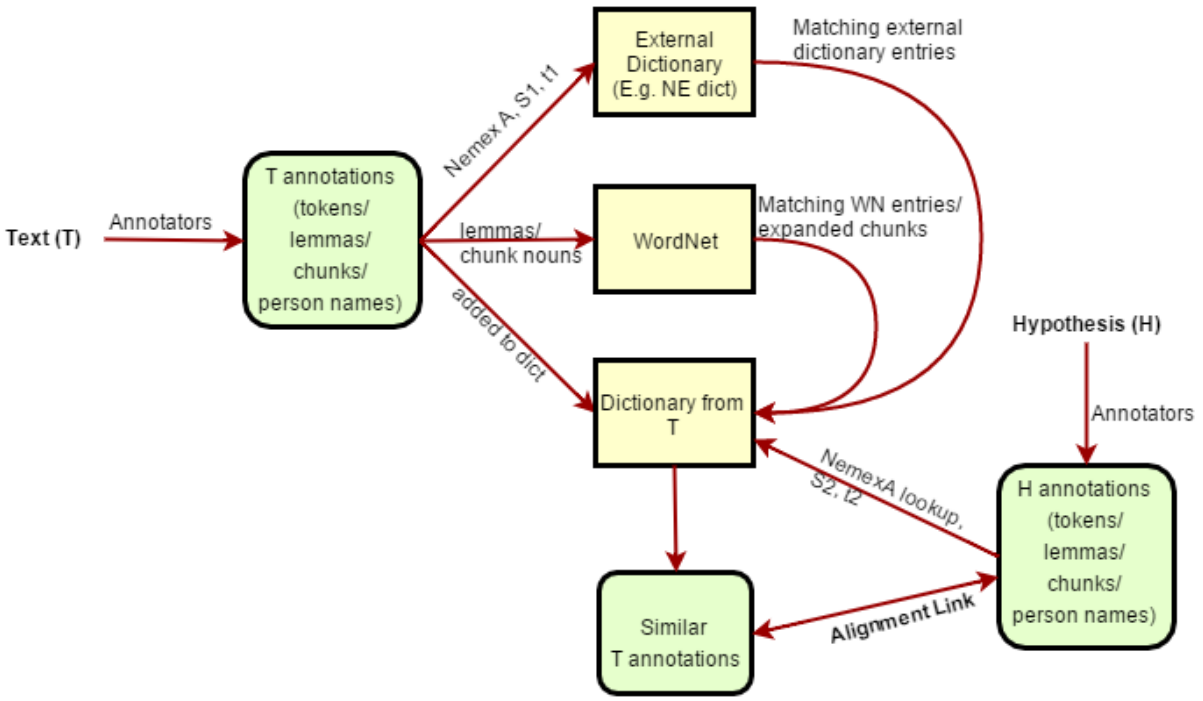


Figure 4. Nemex alignment pipeline (Direction T-to-H)

In the most fundamental setup of these aligners, a gazetteer is first created in the format supported by NemexA, from all relevant entries in T or H (depending on the direction of processing). Spaces in multiword entries are replaced with a prespecified delimiter. During look-up, it can be decided whether the delimiter should be considered or ignored. The entries are processed as a vector of character n-grams of some configured length. Sense, frequency and relative frequency of the entries are also stored in the gazetteer. An inverted list is maintained to identify the offsets in T/H that generate a given entry. If the specified direction of processing is text-to-hypothesis (T-to-H), a gazetteer is generated using annotation entries in the text 'T'. If it is vice-versa, hypothesis-to-text (H-to-T), the gazetteer is instead generated from annotation entries in the hypothesis 'H'. If the direction is left unspecified, "H-to-T" is selected by default.

After obtaining a gazetteer from the 'T'/'H', NemexA similarity calculation functions are queried using entries in 'H'/'T'. All entries in the gazetteer, which are similar to an entry in 'H'/'T', are identified. This similarity calculation is done under a prespecified similarity threshold and distance metric. Thereby, a directional alignment link is added between the queried entry and all the appropriate offsets of the obtained matching entries.

2.4.1 Nemex Bag-of-words Aligner

Nemex bag-of-words aligner, as suggested by its name, performs an alignment between words in text and hypothesis through the use of the tool NemexA. The aligner functions

under its basic settings. The words are the strings corresponding to tokens generated by a tokenizer. A NemexA gazetteer is created from words in 'T'/'H', and the words in the gazetteer similar to all the words in 'H'/'T' are found. Similar words in T and H are aligned under a similarity threshold. An external file containing stop-words in English is available. Stop-words in the content of (T,H) can be ignored by comparing content words with the words in this list.

A bag-of-words exact alignment (similarity threshold 1.0), without stop-words removal, provides a baseline for evaluation.

2.4.2 Nemex Bag-of-lemmas Aligner

Nemex bag-of-lemmas aligner aligns token lemmas in 'T' and 'H'. A lemma refers to the canonical or the base form of a word, without any inflections. An example for such a token lemma is the following:

Words: {*"come", "came", "coming", "comes"*}
Lemma: *"come"*

Using word lemmas instead of exact word forms allows for alignment of different forms of the same word. An alignment should identify corresponding units of text, whether or not meaning the same. Word lemma alignments enable such a match to some extent.

This aligner functions in a similar manner as the Nemex bag-of-words aligner, with a difference being in the entities that are aligned. In addition to the fundamental setup of the aligners, additional semantic alignments are supported through WordNet (Miller, 1995).

Token lemmas, along with their part-of-speech tags, can be looked up in WordNet to expand matching relations. These relations are specified as configurable parameters. According to our hypothesis, for textual entailment to hold, words in 'H' should be synonyms, hypernyms or part-holonyms of words in 'T'. Synonyms are groups of words which have nearly the same meaning. Example:

Synonym pairs: {*'quickly', 'rapidly'*}

Two words 'w1' and 'w2' are said to be hypernyms of each other, if 'w1' is a word whose meaning forms a broad category, and the meaning of 'w2' falls within that category. Example:

Word: *'blue'*
Hypernym: *'color'*

Similarly, 'w1' is said to be a holonym of 'w2' if 'w2' is a part of 'w1'. Example:

***Word**: 'hand'*
***Holonym**: 'body'*

When an entry from 'T'/'H' is added while creating a NemexA gazetteer, all the matching entries from WordNet under the specified relations and word sense are also added to it, with the same offsets as the original entry. During look-up, alignment links based on these additional entries are also added. It results in positive semantic lexical alignments through WordNet.

2.4.3 Nemex Bag-of-chunks Aligner

Nemex bag-of-chunks aligner also performs an alignment similar to approaches mentioned above.

First of all, chunks of text are obtained using the "chunker" from the OpenNLP project. This chunker divides the text into the segments noun phrase, verb phrase, prepositional phrase and other. Part-of-speech (POS) and token annotations are required for this chunker to function. Identified chunks of text serve as the entities that are aligned. This results in a phrase-level alignment of text.

In a NemexA gazetteer, these chunks of text are treated as multi-word entries, where words are separated by a delimiter. The basic aligner works in the same manner as in the other two cases.

WordNet (Miller, 1995) look-up in this aligner is performed in a different manner than the previous one. The nouns in the chunks of text are looked up in WordNet. Multiple entries are generated from each chunk entry by replacing nouns with their corresponding matching entries. Example:

***Chunk**: "the dog and the cat"*
***WordNet queries**: {'dog', 'cat' }*
***Retrieved entries for dog:** {'x','y'}*
***Retrieved entries for cat**: {'p','q'}*
***Finally generated entries**: "the 'l' and the 'm' "; $\forall$ 'l' $\in$ {'dog', 'x', 'y'} and $\forall$ 'm' $\in$ {'cat', 'p', 'q'}*

In a (T,H) pair, length of the hypothesis 'H' is typically more that length of the text 'T'. Due to this property, "H-to-T" direction of processing is preferred in Nemex bag-of-chunks alignments which require WordNet look-up. Generating the gazetteer from H entries instead of T lowers the number of WordNet look-ups. It hence lowers the number of matching synsets, the permutations generated, and thereby the time required for processing.

Given the variability of natural languages, the same information can be expressed in multiple ways. It is highly unlikely that the same content is expressed using the same phrases. In order to align the phrases which are similar, but not necessarily exact, the Nemex bag-of-chunks aligner uses a lower similarity threshold as opposed to the Nemex bag-of-words and Nemex bag-of-lemmas aligner.

2.5 Nemex Scorers

Once Nemex alignments have been generated, various useful features are identified, based on these alignments. For each feature, a score is calculated, which is used during the classification phase. The following scores are calculated for each of the three Nemex aligners:

- **Based on Task setting**: 4 scores are calculated based on the task that the (T,H) pairs have been obtained from - Information Extraction (IE), Information Retrieval (IR), Question Answering (QA) and Summarization (SUM). These scores are 1 if the pair belongs to the corresponding task, and 0 otherwise. If the task information is absent, all these scores are 0. For each instance, maximum one of these four scores is 1.

- **Based on number of alignments**: Multiple features that target the extent of overlap between (T,H) pairs are identified, and corresponding scores are calculated. This is based on the contention that higher the positive overlap between 'T' and 'H', greater is the possibility for entailment. Different alignments generated by the Nemex aligners indicate different levels of positive correspondence between T and H, ranging from surface level to semantic level. The following scores are calculated dependent on the number of generated alignments:

  1. $\frac{|T\&H|}{|T|}$: Number of aligned entities between text and hypothesis, divided by the total number of entities in text. This normalizes the overlap between 'T' and 'H' w.r.t the length of 'T'.
  2. $\frac{|T\&H|}{|H|}$: Number of aligned entities between 'T' and 'H', divided by the total number of entities in 'H'. This normalizes the overlap between 'T' and 'H' w.r.t the length of 'H'.
  3. $\frac{|T\&H|}{|T|} * \frac{|T\&H|}{|H|}$: Product of the previous two scores. This normalizes the overlap between 'T' and 'H' w.r.t the length of the text as well as the hypothesis in a (T,H) pair.

- **Based on content coverage under alignment**: When a phrase-level alignment is performed, the number of alignments is low because the number of phrases in the

content is low. Some of these phrase alignments hold meaningful information. For example, the alignments may indicate whether the actor and the patient in the (T,H) pair are the same; whether the actions described in the text and the hypothesis are contradictions of each other, or do they lead to an entailment relation; and many more similar significant ideas. The following attribute scores are calculated to account for such information:

1. **Coverage of words under alignment**: This score calculates the percentage of words in the hypothesis that are covered by the generated phrase alignments. These words are identified through the token strings segmented by a tokenizer. It is believed that higher the overlap, greater is the possibility for a positive entailment.
2. **Coverage of content words under alignment**: Nouns, adjectives, and verbs in snippets of text indicate the central content in the text. Identification of these terms allows processing significant information. These terms - verbs, nouns and adjectives, are treated as content terms. A score is calculated to find the percentage of content terms in hypothesis that have been positively aligned in a given (T,H) pair. This score is then used as a feature for classification.
3. **Coverage of verbs under alignment**: Verbs denote actions. Comparison of actions taking place in the text and hypothesis provides a reasonable idea about the events discussed in the two. The coverage of verbs in the hypothesis under the alignment of phrases in (T,H) is used as a feature score to decide about textual entailment.
4. **Coverage of proper nouns under alignment**: Proper nouns, or Named Entities (NEs), are significant in comprehending meaning of some text. The subjects and objects, or actors and patients, are usually denoted through 'NE's. The percentage of 'NE's in the hypothesis that have been included under a positive alignment support the decision about a positive entailment, and are used as an attribute score as well.

2.6 Semantic Phrase Aligner

So far, the obtained alignments using Nemex aligners perform a semantic abstraction through the use of WordNet. In this section, we discuss the usage of embedded vectors to find semantic alignments between snippets of text and hypothesis in a (T,H) pair.

An embedded word vector 'v(w)' refers to a distributed representation of a word 'w' with a vector of real numbers. For example, the word "man" may be represented with a vector like:

$$v(man) = [0.0, 0.3, -0.7, 1.0, ...]$$

The size of these vectors typically range from 200 to 500 dimensions. Each dimension is a feature which should ideally capture syntactic and semantic patterns in the data. Higher the number of dimensions, more should be the number of captured linguistic patterns. These vectors can be visualized using tools like t-SNE (Van der Maaten and Hinton, 2008). A sample visualization of embedded vectors is given in Figure 5.

Figure 5. 2D T-SNE word embeddings visualization by Turian et al. (2010) on features generated by Collobert and Weston (2008)

During the development, pretrained word embeddings generated using Word2Vec (Mikolov et al., 2013a,b) have been used. Word2Vec is a tool which generates word vectors using simple architectures that have fast processing speed, allowing for scaling to larger datasets. Word2Vec makes use of two different architectures for training the vectors: the Continuous Bag-of-Words (CBOW) architecture and the skip-gram architecture.

The CBOW architecture generates word vectors through a task which involves prediction of a word given its context. On the contrary, the skip-gram architecture predicts the context terms given a word. For example, in the following

***Sentence**: "I shopped for groceries from the supermarket"*

the **CBOW model** could be trained using the words:

***Input**: {"I", "shopped", "for", "from", "the", "supermarket"}*
***Output**: "groceries"*

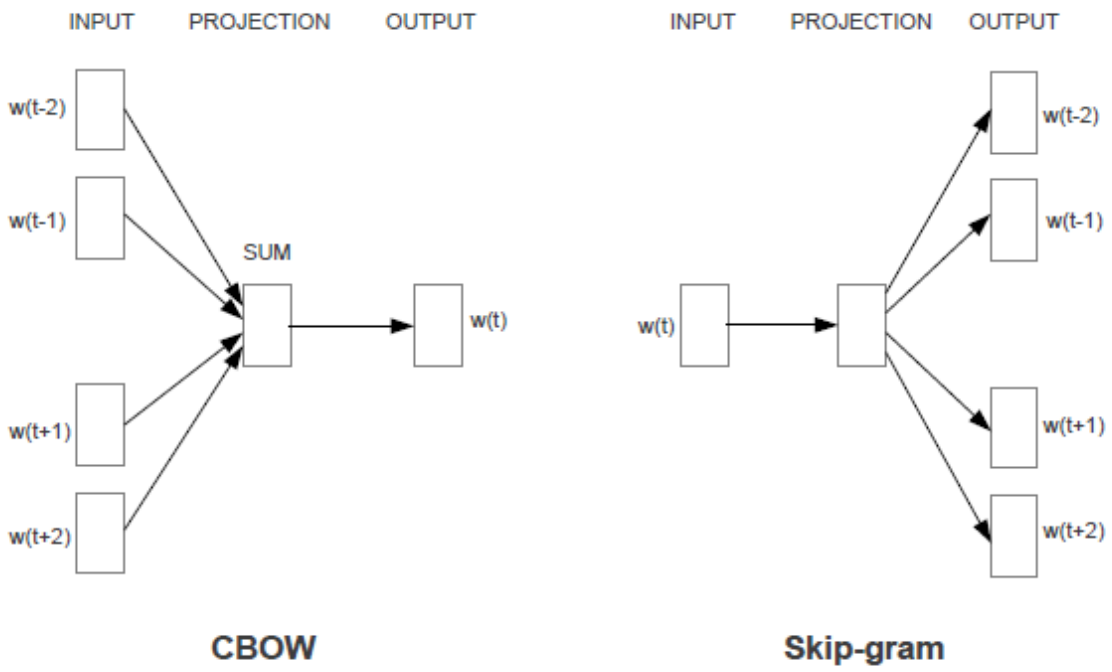


Figure 6. Architecture for Word2Vec CBOW and skip-gram models (Mikolov et al., 2013a)

Similarly, considering the same sentence, the **skip-gram model** could be trained by using:

Input: *"groceries"*
Output: {*"I", "shopped", "for", "from", "the", "supermarket"*}

The skip-gram technique performs better in learning vectors for infrequent terms as opposed to the CBOW architecture.

Vectors are typically learned from a large training corpus. In such corpora, word distribution is highly imbalanced. Little information is gained by co-occurrence of a frequent word like 'a' with infrequent terms like nouns. However, co-occurrence of words like "Neckar" and "river" add a lot of information. In order to learn good vector representations, each word is discarded with a probably computed using a formula,

$$P(w_i) = 1 - sqrt(\frac{t}{f(w_i)})$$

Where t is a threshold, typically around 10^-5, and $f(w_i)$ is the frequency of a word w_i. This sub-samples the frequent terms, preserving well the original frequency rank.

Negative sampling technique is used to further speed up the word2vec training process. In

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Figure 7. Captured relations among trained Word2Vec vectors (Mikolov et al., 2013a)

this technique, only 'k' negative training samples are used corresponding to each positive context of an entity, as opposed to all other negative entities in the vocabulary. It applies a simplified technique of Noise Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2012) for this purpose.

Embedded vectors encode the meaning of a word. Similar words usually lie close to each other in the vector space. It has been shown that these vectors also capture several linguistic associations like notions of gender and tense. Basic mathematical operations on these vectors can be used to assess the learned associations. For example, $vector(\text{“}Delhi\text{”}) - vector(\text{“}India\text{”}) + vector(\text{“}Germany\text{”})$ gives us a vector which lies the closest to the vector for the word *“Berlin”* in the corresponding vector space. Some other examples of the captured associations are given Figure 7.

In order to identify semantic alignments between phrases of 'T' and 'H' in a (T,H) pair in the given data-set, the data-set is first annotated using a Linguistic Analysis Pipeline (LAP) within the EOP (Magnini et al., 2014). Basic tokens and part-of-speech annotations are required for the functioning, which makes basic pipelines like the OpenNLP (Baldridge, 2005) pipeline suitable. Maltparser (Nivre et al., 2006) pipeline also provides the required annotations. Further, chunks of text and hypothesis are obtained using the "chunker" from the OpenNLP project. Thereby, vectors for these chunks, or phrases, are obtained using word vectors. Word vectors used during the development are the vectors trained on a corpus obtained from Google News, consisting of about 100 billion words. The corresponding vocabulary size for this corpus is 3 million, where the entities contain both words and automatically derived phrases. The CBOW architecture was used for training, with sub-sampling using threshold 1e-5, and with negative sampling with 3 negative examples per each positive one. The number of dimensions of the generated vector set is 300.

Indian + actor	**Swiss + currency**	**Russian + jet**	**German + drink**
actor_Kabir_Bedi	swiss_franc	Cessna_Citation_XLS	###ml_glasses
Aamir_Khan_Shahrukh_Khan	curency	Yakovlev_Yak	Franziskaner
Abhishek_Bachan	Russian_Rouble	Citation_Encore	Glenlivet_Scotch
Ananth_Nag	Forint	turboprop_airliner	+_Hanzo
filmmaker_BR_Chopra	economies	Aeroflot	ounce_sugar_sweetened

Table 1. Top five closest terms for given queries

Basic arithmetic operations can be performed on word vectors to obtain meaningful results. Vectors for phrases are calculated using a composition of individual vectors of each word in that phrase. The composition operator used for this calculation is the basic 'summation' operator. Table 1 shows the closest five terms obtained on querying words joined with addition operator in the Google News vectors.

During the calculation of phrase vectors, all the words except proper nouns are converted to lowercase. An identification for proper nouns is performed using part-of-speech tags that have been added by the annotation pipeline. The words which are not present in the vocabulary of the Google News corpus, and the punctuation symbols are ignored. Vectors for the rest of the words are added. Stop-words have not been removed during this calculation. For example, in the noun phrase, *"Chris, Bill, and the cat's Owner"*, if "Chris" is not present in the vocabulary, the vector will be calculated as follows:

$vector(\text{“}Chris\ and\ the\ cat's\ owner\text{”}) = vector(\text{“}Bill\text{”}) + vector(\text{“}and\text{”}) + vector(\text{“}the\text{”}) + vector(\text{“}cat\text{”}) + vector(\text{“}owner\text{”})$

After calculating this sum, the resulting vector is normalized to find a final phrase vector. To make training and testing faster, these vectors for the complete data-set are calculated in advance and are input directly for generating alignments.

Once the vectors for phrases in 'T' and 'H' have been obtained, a similarity between them is calculated in a brute-force manner to generate all possible alignments. A cosine distance is used as a similarity metric, which is calculated as dot product of normalized vectors. If the cosine distance is greater than a prespecified threshold, an alignment link between the phrases is added.

In the other alignment techniques discussed earlier, alignment links are added only between the entities which indicate a positive entailment. A Nemex match without WordNet only returns positively similar relations. Within the Nemex aligners, WordNet is used to query only such relations which indicate entailment. However, when semantic alignments are obtained using embedded word vectors, such controls are lost. Similarity between vectors indicates a similarity between the usage of corresponding entities in some context. These similarities may be positive or negative with respect to an entailment relation. For

example, there may be a generated alignment link between the phrases "is conservative" and "is liberal", given that the similarity threshold permits this.

In order to distinguish between these positive and negative alignment links, usage of the knowledge bases WordNet (Miller, 1995) and VerbOcean (Chklovski and Pantel, 2004) has been proposed. Words in the aligned pair of phrases are inspected for negative relations among them. These negative relations refer to those relations which indicate a non-entailment or a contradiction relation. Examples of such relations are antonym pairs in WordNet, and those relations in VerbOcean, where the hypothesis terms are stronger-than or opposite of the terms in the text. Antonyms are sets of words, whose meanings are opposite of each other. "Antonym" relation in WordNet is equivalent to the "opposite-of" relation in VerbOcean. Example:

Antonyms: {*open, close*}

Similarly, the "stronger-than" relation is defined between two words 'w1' and 'w2', such that the strength of 'w1' is more than that of 'w2'. Example,

T term: {*stab*}
H term: {*kill*}
Relation: {*H is stronger-than T*}

Any alignment link between a pair of phrases is deemed to be negative, if a negative relation between any pair of words between the two is identified. For example, if the following two phrases were aligned using an embedded chunk vector alignment scheme,

T phrase: *"has been stabbed"*
H phrase: *"has been killed"*

Because "killed" is stronger-than "stabbed", the phrases would hold a negative alignment. Similarly, the following two pairs would be negatively aligned because the words {*shout,whisper*} are antonyms of each other:

T phrase: *"is shouting"*
H phrase: *"has been whispering"*

The pipeline for a semantic alignment between vectors of phrases using word embeddings is elaborated in Figure 8.

2.7 Semantic Phrase Scorer

Once alignments have been generated between phrases in text and hypothesis using embedded word vectors trained by Word2vec (Mikolov et al., 2013a), various features for en-

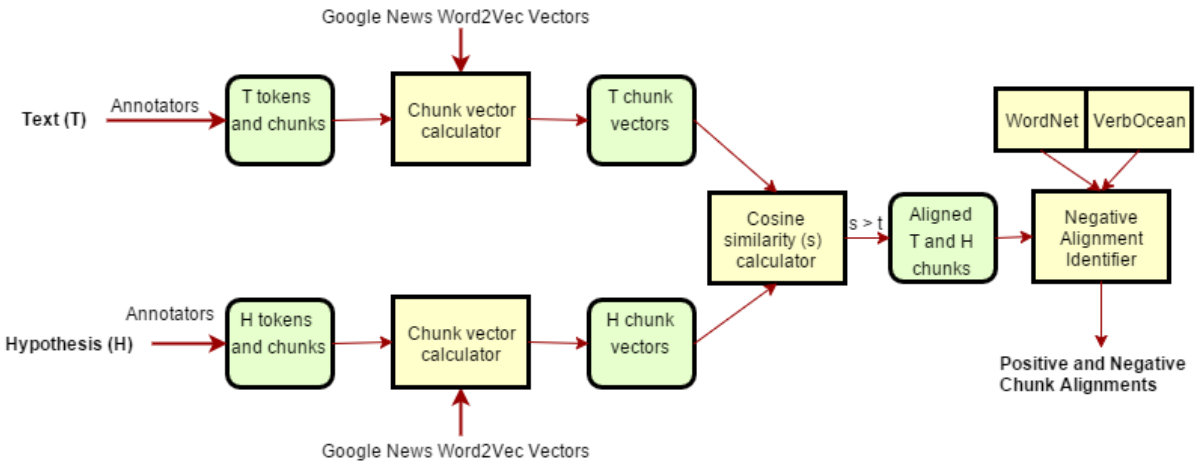


Figure 8. Embedded chunk vector alignment pipeline

tailment classification are identified and scores are calculated. These features and scores are similar to those discussed in section 2.5.

As discussed in section 2.6, these alignments between phrases are categorized as positive and negative alignments. Positive alignments refer to those alignment links, in which pair of phrases hold similar meaning and may contribute towards a positive entailment. Similarly, negative alignments refer to those alignment links in which phrases may have contradictory meanings, or meanings which may contribute towards a non-entailment.

To train a classifier for a binary entailment decision task, following scores are calculated in the proposed system based on the semantic phrases alignments identified through the use of embedded word vectors:

1. **Number of positive alignments with respect to hypothesis**: This feature is developed on the presumption that more the number of phrases in hypothesis that are similar in meaning to some phrase in text, higher are the chances for a positive entailment. A score is calculated as $\frac{|positive\ T\&H\ phrase\ alignments|}{|H\ phrases|}$.

2. **Number of negative alignments w.r.t hypothesis**: This feature is associated with a contention that presence of phrases in hypothesis that mean opposite to or imply stronger actions than corresponding content in text result in a higher possibility of a non-entailment relation. Score is calculated as the fraction - $\frac{|negative\ T\&H\ phrase\ alignments|}{|H\ phrases|}$.

3. **Coverage of content under positive alignments**: Multiple scores are calculated based on the coverage of terms like words, content words, verbs and proper nouns by positive alignment links between 'T' and 'H', in the same manner as discussed in Nemex scoring techniques in Section 2.5. The difference between coverage scores for the "Nemex bag-of-chunks aligner" and the "Embedded chunk vector aligner"

is that only positively aligned links are used in the latter case. Nemex aligners add only positive alignment links to the (T,H) pairs, which makes this distinction irrelevant.

2.8 Negation Scorer

While identifying entailment relations, some terms are more significant than other terms in a (T,H) pair. These terms alter the meaning of a statement significantly. Some of such terms are negation words like “no”, “not”, “none”, “neither”, etc. Non-uniform distribution of such words across the text and hypothesis often result in contradictions. For example,

***T**: Spain did not win the championship.*
***H**: Spain won the championship.*
***Relation**: Non-entailment (contradiction)*

Similarly, if the hypothesis contains an antonym of a word in the text, along with a negation word around it, the meaning is frequently inverted back to the meaning of the text. For example,

***T**: I like beer.*
***H**: I do not hate beer.*
***Antonym pairs**: {like, hate}*
***Negation words in H**: not*
***Relation**: Entailment*

Calculation of a negation score while training an entailment decision classifier is important due to such patterns. In the proposed system, the relative distribution of negation words in ‘T’ and ‘H’ has been used to calculate such a score.

First, the data is tokenized using any of the linguistic annotation pipelines supported by the EOP. Thereby, obtained token strings are matched against a set of negation words, which are input as an external file. A counter for negation terms is increased every time a positive exact match is found. Once the number of negation terms in both text and hypothesis have been obtained, a negation score is calculated as follows:

- $tNeg = \frac{number\ of\ negation\ words\ in\ T}{number\ of\ tokens\ in\ T}$
- $hNeg = \frac{number\ of\ negation\ words\ in\ H}{number\ of\ tokens\ in\ H}$
- If *tNeg* is zero, negation score is the value of *hNeg*.
- Otherwise, if *hNeg* is zero, negation score is the value of *tNeg*.

- Otherwise, negation score is the fraction $\frac{hNeg}{tNeg}$.

In Section 2, different approaches to generate alignments between entities in text and hypothesis were discussed. The working of Nemex bag-of-words, Nemex bag-of-lemmas and Nemex bag-of-chunks aligner, using the NemexA tool developed at DFKI, Saarbrücken, was elaborated along with techniques to calculate scores for different features using these alignments. Furthermore, techniques were discussed to semantically align phrases in text and hypothesis using word vectors generated by Word2Vec. Among these alignments, positive and negative alignments were distinguished between, and various scores were calculated for features related to entailment classification. Moreover, a negation score was calculated to account for drastic changes in meaning due to the use of negation terms like "no". Using the calculated scores, an architecture which performs a logistic regression classification using L2 regularization was proposed to train an entailment classification model.

In the next section, Section 3, the evaluations performed and the results obtained have been discussed.

3 Evaluations and Results

3.1 Data-sets and Evaluation Metrics

The proposed EDA has been evaluated on three standard data-sets in English language, which have been introduced earlier in sections 1.3.1 and 1.3.6 - the RTE 3 and 6 data-sets from RTE challenges, and the SNLI corpus. Standard evaluation metrics from the RTE challenges have been used to assess the performance of the developed algorithms and compare it with other equivalent systems. In this section, these data-sets will be described along with the corresponding evaluation metrics used.

3.1.1 RTE-3 data-set

The English data-set from the third challenge in the series of Recognizing Textual Entailment challenges, RTE-3, consists of 800 pairs of text and hypothesis for development and testing each, balanced between two labels - entailment and non-entailment. 200 pairs have been obtained from each of the following Natural Language Processing (NLP) tasks: information extraction, information retrieval, question answering and multi-document summarization. The pairs have been extracted from freely available sources like WikiNews and Wikipedia, and are mostly outputs of web based systems for the given tasks.

Text (T) in the (T,H) pairs in the data-set consists of longer content, constituting up to a paragraph. However, the corresponding hypothesis (H) is typically just a sentence. Longer texts pose a requirement for discourse analysis. Moreover, the texts (T) may consist of imperfect grammar or style, where minor spelling and punctuation errors have been corrected. However, the hypotheses (H) have been cross-verified by a native English speaker.

Entailment decisions in the data-set assume the availability of common world knowledge. In case of partial entailment of a hypothesis by a text, a non-entailment judgment is provided. Highly probable, but not completely certain cases are assumed to be positively entailing cases.

Two sample pairs from the RTE-3 data-set are the following:

***T**: "The Extra Girl" (1923) is a story of a small-town girl, Sue Graham (played by Mabel Normand) who comes to Hollywood to be in the pictures. This Mabel Normand vehicle, produced by Mack Sennett, followed earlier films about the film industry and also paved the way for later films about Hollywood, such as King Vidor's "Show People" (1928).*
***H**: "The Extra Girl" was produced by Sennett.*

***Entailment class**: Entailment*
***Task**: Information Extraction*

***T**: Stolen Warhol works recovered: Amsterdam police said Wednesday that they have recovered stolen lithographs by the late U.S. pop artist Andy Warhol worth more than $1 million. Dali's paintings are still missing.*
***H**: Millions of dollars of art were recovered, including works by Dali.*
***Entailment class**: Non-entailment*
***Task**: Summarization*

An accuracy percentage is used to evaluate the performance of the proposed algorithms on the RTE-3 data-set. Accuracy refers to the fraction of correct classifications, with respect to total number of instances. For a binary classification task, accuracy is defined as,

$$Accuracy = \frac{True\ Positive + True\ Negative}{total\ no.\ of\ instances\ in\ test\ set}, \tag{10}$$

where true positive and true negative are defined as given in table 2.

	True class A	**True class not A**
Predicted class A	True Positive (TP)	False Positive (FP)
Predicted class not A	False Negative (FN)	True Negative (TN)

Table 2. Performance table where instances have class label A

3.1.2 RTE-6 data-set

English data-set from the sixth Recognizing Textual Entailment challenge, RTE-6, consists of a corpus and a list of hypotheses, instead of a set of (T,H) pairs. The task is to find all the sentences in the corpus that entail a given hypothesis. The data-set has been developed in a summarization setting. Corpora have been selected from a summarization task data-set, and most of the hypotheses are sentences from this data-set that have been included in automatic summaries generated by summarization systems. However, to support a pairwise approach, a list of sentences retrieved using Apache Lucene for each hypothesis is also provided. The entire data-set is divided into 20 topics - 10 each for development and testing. It is organized as follows: there are two clusters of documents - cluster A with 10 documents from an earlier publication date and cluster B with 10

documents from a later date. There are up to 30 hypotheses for each topic from cluster B documents, and up to 100 candidate sentences for each hypothesis from cluster A documents. The hypotheses have been rephrased as independent sentences by resolving discourse references and applying minor syntactic and morpho-syntactic changes. However, the candidate text sentences have been left unchanged.

This organization of data-set generates 15,955 (T,H) pairs in the development set, out of which 897 pairs result in a positive entailment of hypothesis by text. Similarly, there are a total of 19,972 generated (T,H) pairs in the test set, out of which 945 are positive entailment relations. 89 hypotheses in the development set, and 100 in the test set do not have any corresponding entailing text sentences.

The data-set reflects entailment distribution in a natural setting, which consists of only about 5% of positive cases. This presents all the challenges faced while identifying entailment relations in a real task. One example of such a challenge is discourse identification in a text corpus. All implicit references to previously mentioned date, time, event, entities etc. in the given topic need to be resolved. Publication date of the document should also be tracked for time and tense resolution.

To evaluate system performance on the RTE-6 data-set, micro-averaged F-score is calculated on the entailing sentences retrieved by the system for the entire test corpus (composed of 10 topics), compared with the entailing sentences in the gold standard. Further, a macro-averaged F-score is also calculated by calculating precision, recall and F-score for each topic separately. Only positively entailing cases are considered for evaluation.

First of all, precision, recall and F-score are defined in the following paragraph. Thereby, micro and macro-averaged precision, recall and F-score are explained.

For this entailment decision task, precision refers to the fraction of correct positive classifications, out of all the sentences that have been classified as positively entailing the hypothesis. It is defined as:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{11}$$

Similarly, recall refers to the fraction of correct positive classifications, out of all the positively entailing sentences present in the corpus. It is defined as:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{12}$$

Based on precision and recall, an F-score is calculated as follows:

$$F - score(\beta) = \frac{(1+\beta^2) * Precision * Recall}{\beta^2 * Precision + Recall}, \tag{13}$$

where β is chosen as 1 unless explicitly specified otherwise.

Now, micro-average precision, recall and F-scores are defined as follows:

Micro-average precision refers to the fraction,

$$micro\ average\ precision = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FP_i)}, \tag{14}$$

where n is the number of topics in the test-set, TP_i is the number of true positives for topic i, and FP_i is the number of false positives for topic i.

Similarly, n and TP_i being the same as before, micro-average recall is calculated as,

$$micro\ average\ recall = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FN_i)}, \tag{15}$$

FN_i is the number of false negatives for topic i.

Micro-average F-score is calculated as in Equation 13 using micro-averaged precision and recall values instead.

Furthermore, macro-average precision, recall and F-score across topics are defined below.

$$macro\ average\ precision = \frac{1}{n}\sum_{i=1}^{n} P_i, \tag{16}$$

such that n is the total number of topics in the test-set, and P_i is precision for topic i.

$$macro\ average\ recall = \frac{1}{n}\sum_{i=1}^{n} R_i, \tag{17}$$

with n being the same as before, and R_i being recall for topic i.

Macro average F-score is calculated in the same manner as Equation 13 using macro-averaged precision and recall values.

Corresponding percentage values of these scores are recorded for comparison with scores of state-of-the-art systems.

3.1.3 Stanford Natural Language Inference corpus

The Stanford Natural Language Inference (SNLI) corpus is a very new and a very large corpus in English language for identifying entailment relations. It consists of 550,152 pairs of text and hypothesis for training, and 10,000 pairs each for development and testing purposes. The data-set has been manually annotated for a balanced composition for a three-way entailment task - entailment, contradiction and semantic independence (neutral). In the cases where annotators could not agree on a class, a '-' label has been provided. Entity and event co-references have been controlled to a great extent during corpus development, which eliminates the requirement for discourse identification. Hypotheses length in the corpus averages to 8.3 tokens, being shorter than texts which contain 14.1 tokens on average. Moreover, majority of sentences in the data-set are syntactically complete.

A few sample (T,H) pairs from the SNLI corpus is given below:

***T**: This church choir sings to the masses as they sing joyous songs from the book at a church.*
***H**: The church is filled with song.*
***Entailment class**: Entailment*

***T**: This church choir sings to the masses as they sing joyous songs from the book at a church.*
***H**: The church has cracks in the ceiling.*
***Entailment class**: Neutral*

***T**: This church choir sings to the masses as they sing joyous songs from the book at a church.*
***H**: A choir singing at a baseball game.*
***Entailment class**: Contradiction*

This corpus is two orders of magnitude larger than other available resources for textual entailment, and has been developed to promote the usage of machine learning techniques in the field of Natural Language Inference (NLI).

Performance of the developed algorithms on the SNLI corpus has been evaluated using the same standards as the RTE-3 data-set: accuracy on a two-way entailment decision task. "Neutral" and "contradiction" labels are both considered as non-entailment relations for this evaluation.

3.2 Evaluations Performed and Results

In this section, results corresponding to evaluations performed on different configurations of the developed classification algorithm will be discussed. During all the evaluations, a standard configuration of the NemexA tool has been used. This configuration uses n-grams of 3 characters to generate gazetteer entry and query vectors without ignoring duplicate n-grams. '#' is used as a multi-word separator. However, similarity measure and threshold have been varied during the process. These similarity measures and thresholds have been specified while describing each experiment.

3.2.1 Evaluations - RTE3 data-set

Multiple evaluations have been performed on English RTE-3 data-set to find an optimal configuration for the system. This optimal configuration is thereby used to evaluate system performance on other data-sets.

In this section, results for these evaluations have been discussed. First, the baseline results for the developed system on RTE-3 data-set have been recorded. Further, effect of pre-processing the data-set by stop-words removal has been shown. Thereafter, effect of approximation on entailment classification accuracy has been discussed for all the supported scorers. Differences between T-to-H and H-to-T directions of processing has also been shown here. Later, effect of coverage features has been analyzed for phrase alignments generated through Nemex scorers and word embeddings. In the end, the best obtained accuracy has been compared with that of the state-of-the-art systems.

Table 3 lists the baseline accuracy for the developed system on RTE3 data-set. This baseline configuration is an exact Nemex bag-of-words scoring, without any pre-processing like stop-words removal. Accuracy for all the supported distance metrics have been recorded. Results for both the directions of processing, T-to-H and H-to-T, have been compared.

S.No.	Similarity Measure	Direction	Accuracy
1	Cosine	T-to-H	60.50
2	Dice	T-to-H	60.63
3	Jaccard	T-to-H	60.63
4	Cosine	H-to-T	60.75
5	Dice	H-to-T	60.25
6	Jaccard	H-to-T	60.50

Table 3. Baseline accuracy: Exact Nemex Bag-of-words scoring without stop-words removal

S.No.	Similarity Measure	Direction	Accuracy
1	Cosine	T-to-H	63.00
2	Dice	T-to-H	62.88
3	Jaccard	T-to-H	63.25
4	Cosine	H-to-T	63.38
5	Dice	H-to-T	63.13
6	Jaccard	H-to-T	63.25

Table 4. Accuracy: Exact Nemex bag-of-words scoring without stop-words

These baseline accuracy are different for different similarity measures. This is so because similarity metrics are defined differently. Similarity value changes with a change in the similarity measure being used.

Moreover, accuracy also varies on varying direction of processing. This is due to the manner in which scores for different features are calculated. If a NemexA gazetteer is generated from words in a hypothesis (H), the maximum number of alignment links would be the number of words in the corresponding text (T). However, if this gazetteer is created from words in a text (T) instead, the maximum number of alignment links that can be added would be the number of words in the corresponding hypothesis (H). Difference in number of alignments results in different values for calculated feature scores.

Next, in Table 4, effect of stop-words removal under an exact Nemex bag-of-words match has been shown. Thereby, this effect has been visualized in Figure 9.

At least 2% increase in accuracy is observed on removing stop-words. This shows that presence of very frequent terms in data results in a loss of meaningful information when entailment classification is performed using bag-of-words scoring. Following this analysis, stop-words have been removed for further evaluations.

Next, the effect of a distance based match, as opposed to an exact match, has been shown. NemexA similarity threshold has been varied for different distance metrics, and corresponding accuracies on the RTE-3 test-set have been recorded. These effects have been shown under the configuration Nemex bag-of-words scoring, when stop-words have been removed. Table 5 lists the accuracies obtained.

Graphs in Figures 10 and 11 visualize the effect of approximation on accuracy of classification of instances in RTE-3 test-set, with respect to directions of processing T-to-H and H-to-T respectively.

As it can be seen from the plots, different similarity measures show similar trend of change in accuracy on varying similarity threshold, for both the directions of processing. Accuracy improves on decreasing the similarity threshold. However, beyond a certain

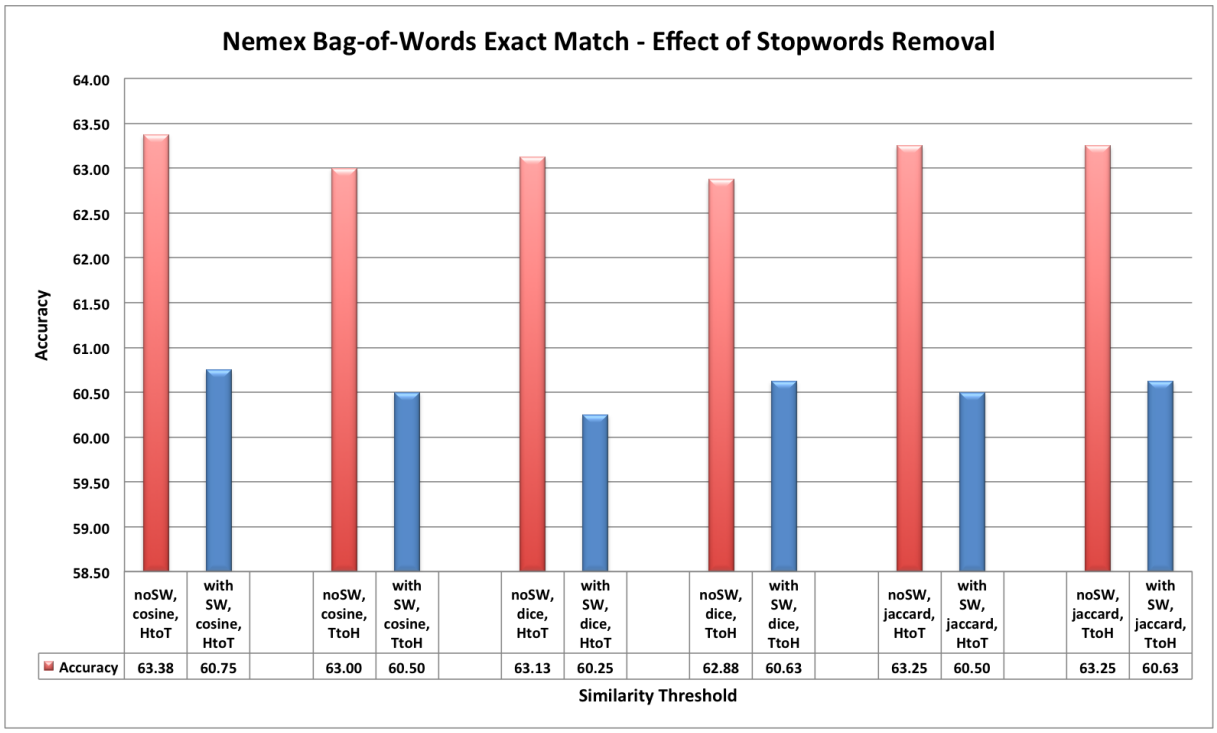


Figure 9. Effect of stop-words removal on exact bag-of-words scoring

Threshold	Dice TtoH accuracy	Dice HtoT accuracy	Cosine TtoH accuracy	Cosine HtoT accuracy	Jaccard TtoH accuracy	Jaccard HtoT accuracy
0.70	64.63	64.50	64.50	64.38	65.00	64.50
0.75	64.63	64.50	64.63	64.63	64.50	64.75
0.80	64.63	64.88	64.88	64.63	64.13	64.00
0.85	64.75	64.75	64.50	64.63	63.75	63.63
0.90	63.75	64.13	63.75	63.75	63.25	63.50
0.95	63.25	63.25	63.25	63.38	63.13	63.00
1.00	62.88	63.13	63.00	63.38	63.25	63.25

Table 5. Effect of distance based match on Nemex bag-of-words scoring after stop-words removal

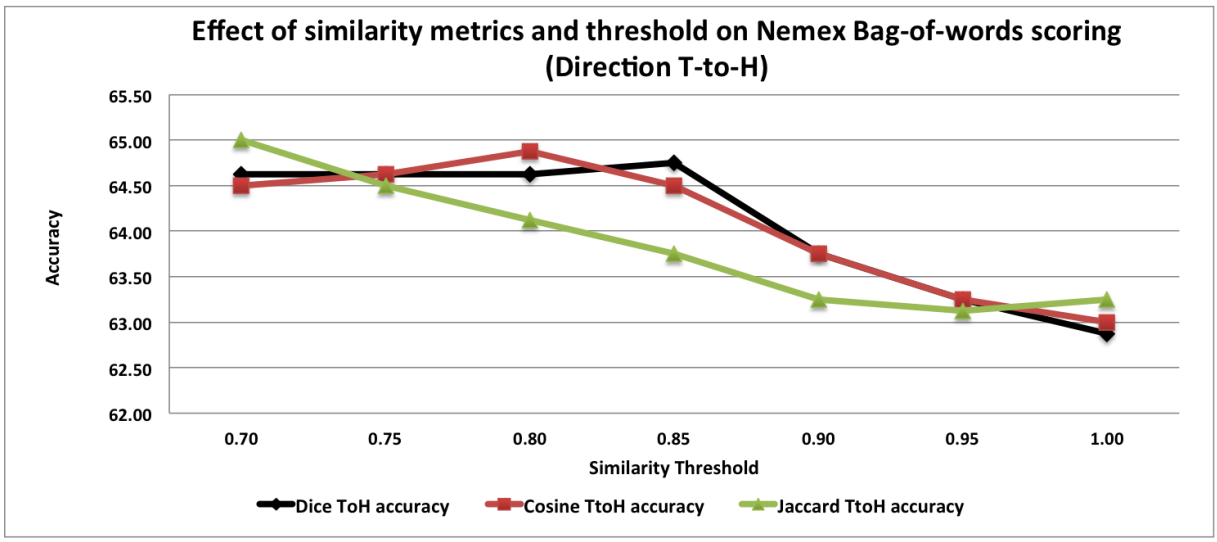


Figure 10. Nemex bag-of-words scoring: effect of approximation, direction T-to-H

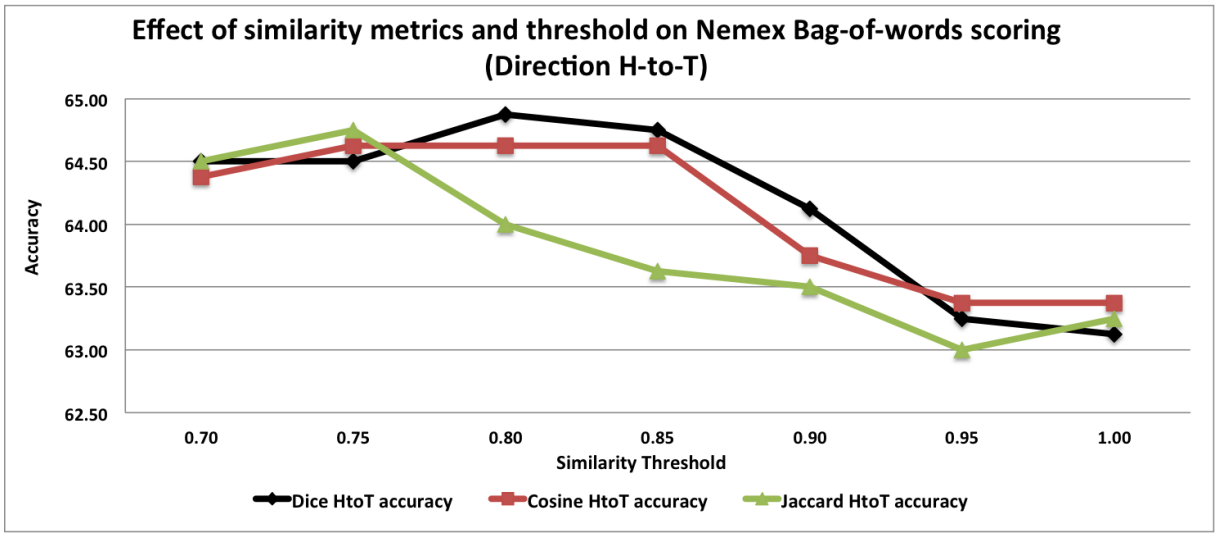


Figure 11. Nemex bag-of-words scoring: effect of approximation, direction H-to-T

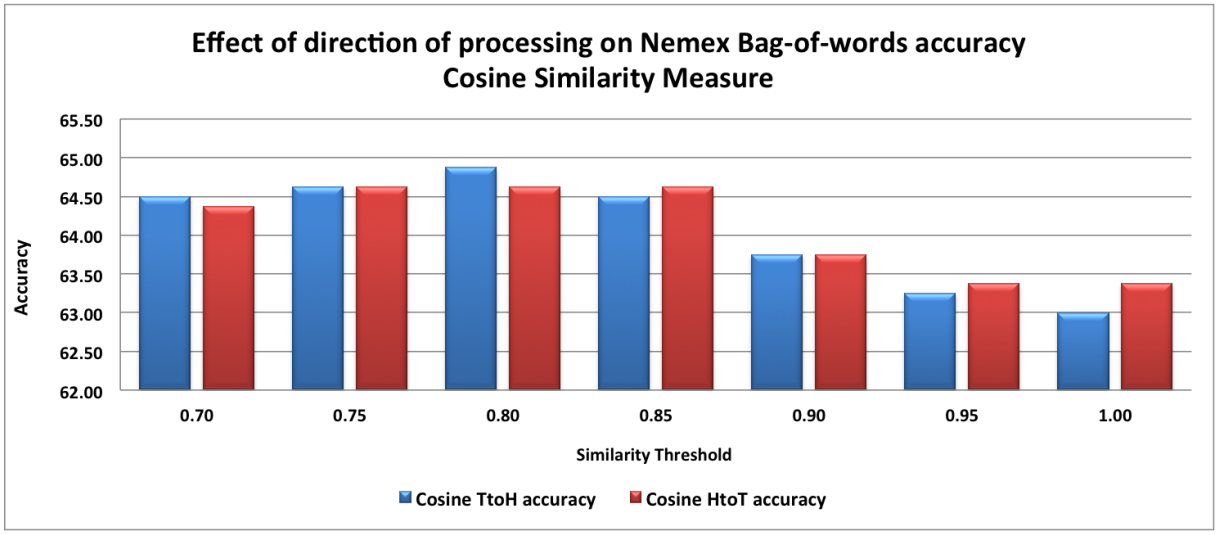


Figure 12. Effect of direction of processing: Nemex bag-of-words scoring, Cosine similarity

value, a decrease in threshold leads to a lower accuracy. This pattern is observed because a distance-based match allows accounting for different forms of the same word, tokenization errors and spelling variations. However, decreasing it beyond a certain value results in addition of noise, leading to a drop in accuracy.

Accuracy obtained using Jaccard similarity is different from those of Dice and cosine similarity metrics. This is because the Jaccard similarity metric penalizes a low overlap between 'T' and 'H' more than Dice and cosine similarity measure. Dice and cosine similarity measures are identical if the vector size for 'T' and 'H' are the same. If overlap between the two vectors is low, cosine similarity value is higher than Dice similarity.

Next, Figures 12, 13 and 14 show the change in accuracy on varying the direction of processing, for Dice, cosine and Jaccard similarity metrics respectively. No clear trend is observed with this variation. However, the values for each direction of processing is different, following the same contention as earlier.

Further, evaluations have been performed using the configuration Nemex bag-of-lemmas scoring, ignoring stop-words in the data-set. Multiple similarity metrics and thresholds have been used to identify classification accuracy trends and best performance setting. Corresponding results have been recorded in Table 6 and have been visualized in the line plots in Figures 15 and 16.

Distance-based alignment follows similar trend in case of a Nemex bag-of-lemmas match as in Nemex bag-of-words scoring for Cosine and Dice similarity measures. Variations in accuracy are less steep as compared to the previous case. This is because in case of Nemex bag-of-words scoring, an approximation allows for alignment of different forms of the same word also, which are already aligned in case of exact Nemex bag-of-lemmas

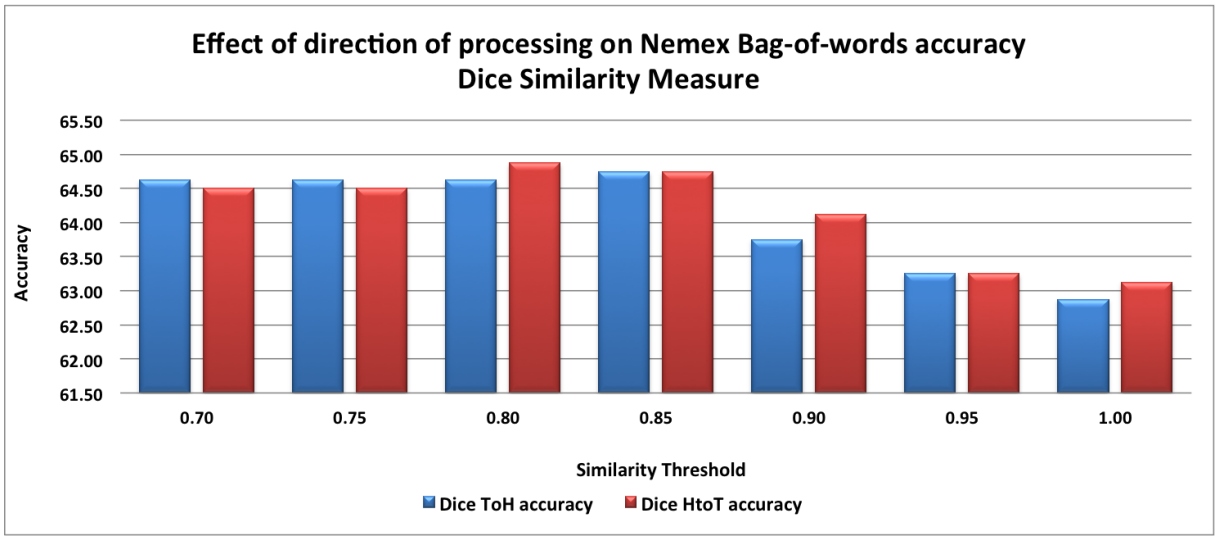


Figure 13. Effect of direction of processing: Nemex bag-of-words scoring, Dice similarity

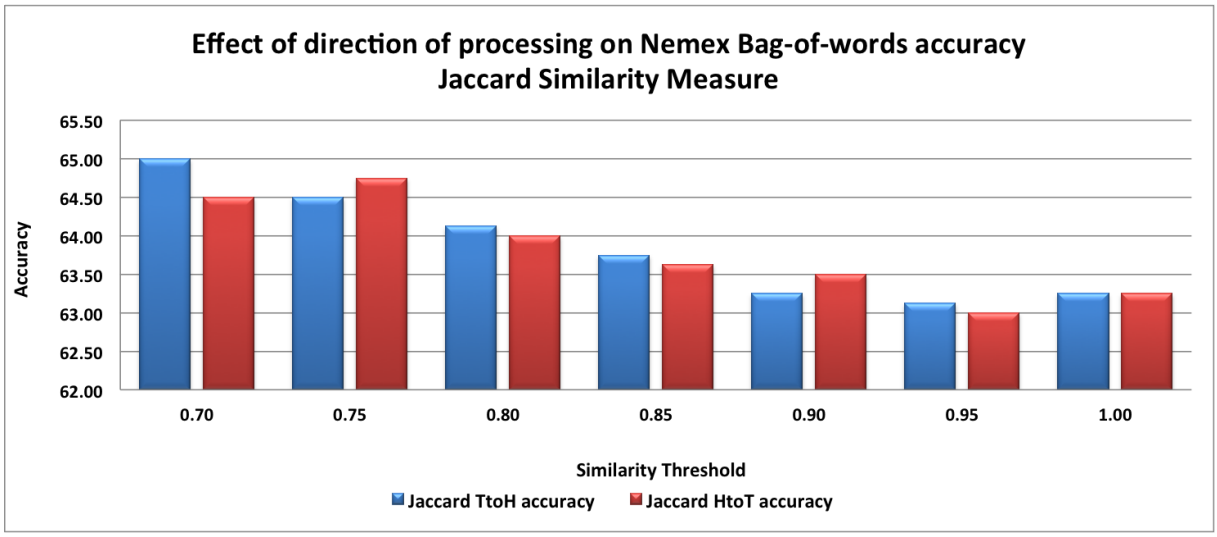


Figure 14. Effect of direction of processing: Nemex bag-of-words scoring, Jaccard similarity

Threshold	Dice TtoH accuracy	Dice HtoT accuracy	Cosine TtoH accuracy	Cosine HtoT accuracy	Jaccard TtoH accuracy	Jaccard HtoT accuracy
0.70	58.88	61.38	60.88	61.75	58.50	57.38
0.75	62.00	60.88	61.25	61.13	58.25	55.88
0.80	58.88	58.63	58.88	58.38	57.13	56.25
0.85	57.00	55.25	57.75	55.75	56.00	55.38
0.90	56.50	55.88	57.13	55.88	56.25	54.88
0.95	56.13	54.88	56.63	55.13	56.88	54.63
1.00	56.13	54.75	56.25	55.00	56.13	54.50

Table 6. Effect of distance based match on Nemex bag-of-lemmas scoring after stopwords removal

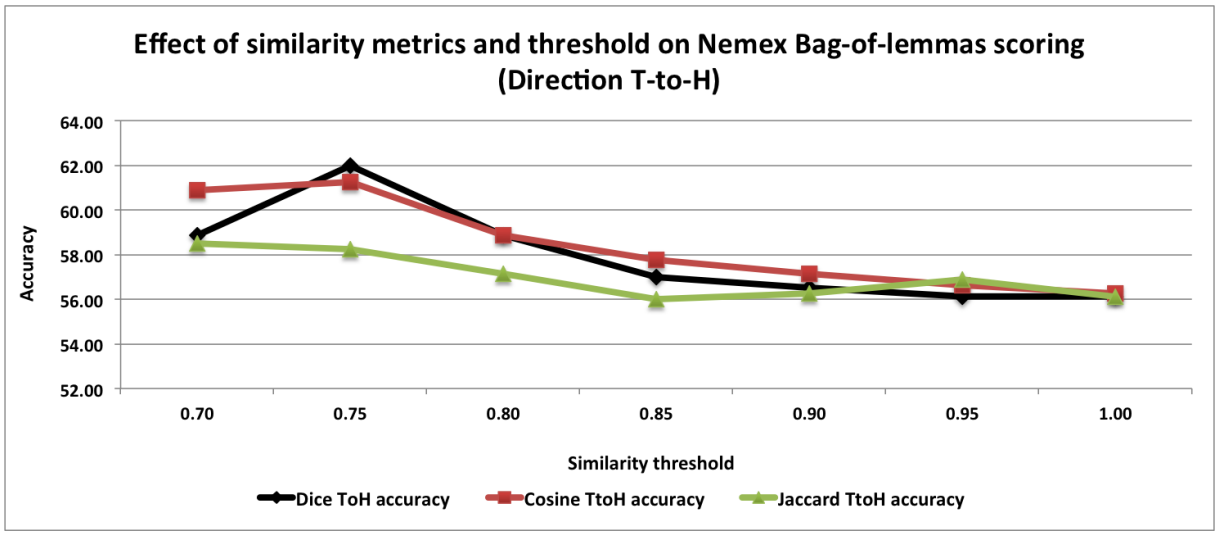


Figure 15. Nemex bag-of-lemmas scoring: effect of approximation, direction T-to-H

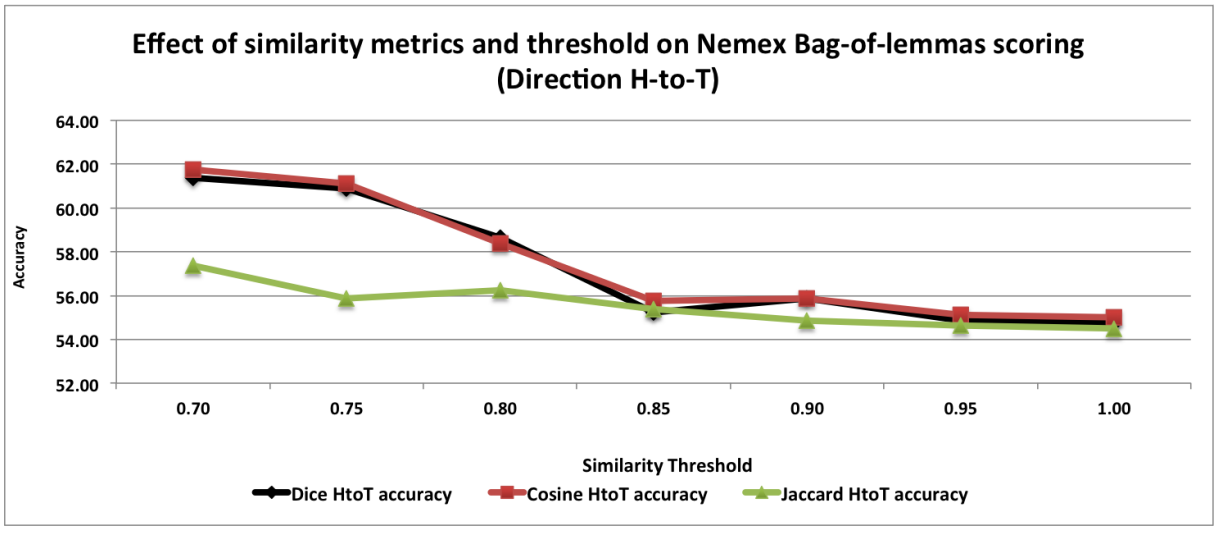


Figure 16. Nemex bag-of-lemmas scoring: effect of approximation, direction H-to-T

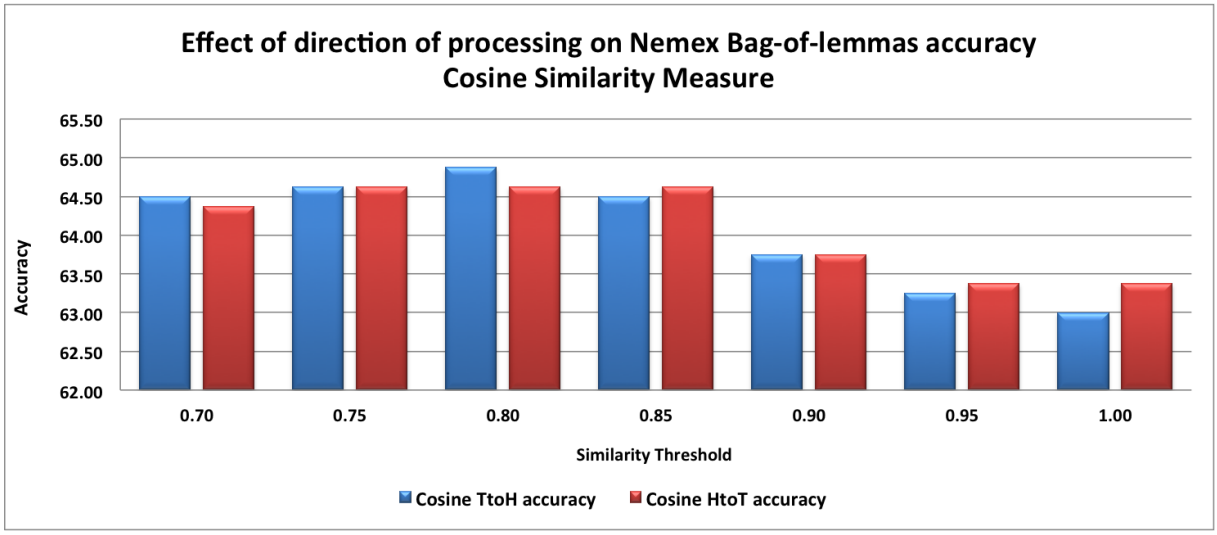


Figure 17. Effect of direction of processing: Nemex bag-of-lemmas scoring, Cosine similarity metric

WordNet Present	Accuracy (Dice, 0.75)	Accuracy (Dice, 0.90)	Accuracy (Cosine, 0.8)
No	62.00	56.50	58.88
Yes	60.88	54.13	58.00

Table 7. Effect of WordNet entries on Nemex bag-of-lemmas scoring

scoring.

Moreover, Nemex bag-of-lemmas scoring performs less accurate entailment classification than Nemex bag-of-words scoring. This suggests that loss of semantics in data due to loss of inflection is high enough to negatively affect system performance.

Following the analysis of the effect of approximation on Nemex bag-of-lemmas scoring, the effect of change in direction of processing on accuracy has been visualized for the same configuration. Figures 17, 18 and 19 represent these visualizations for Cosine, Dice and Jaccard similarity measures respectively. As in the case of Nemex bag-of-words scoring, no clear trend is observed on changing the direction.

Next, evaluations were performed by enriching the generated NemexA gazetteer for Nemex bag-of-lemmas scoring with matching entries (synonyms, hypernyms and part-holonyms) from WordNet. The best performing configuration from the Nemex bag-of-lemmas scoring, and some other randomly chosen configurations were evaluated to analyze this effect. Obtained results have been recorded in Table 7.

It is observed that additional WordNet entries negatively affect the system performance, instead of improving classification accuracy. It indicates that out-of-context entries are perhaps added to the generated gazetteer through a WordNet match, which results in noisy

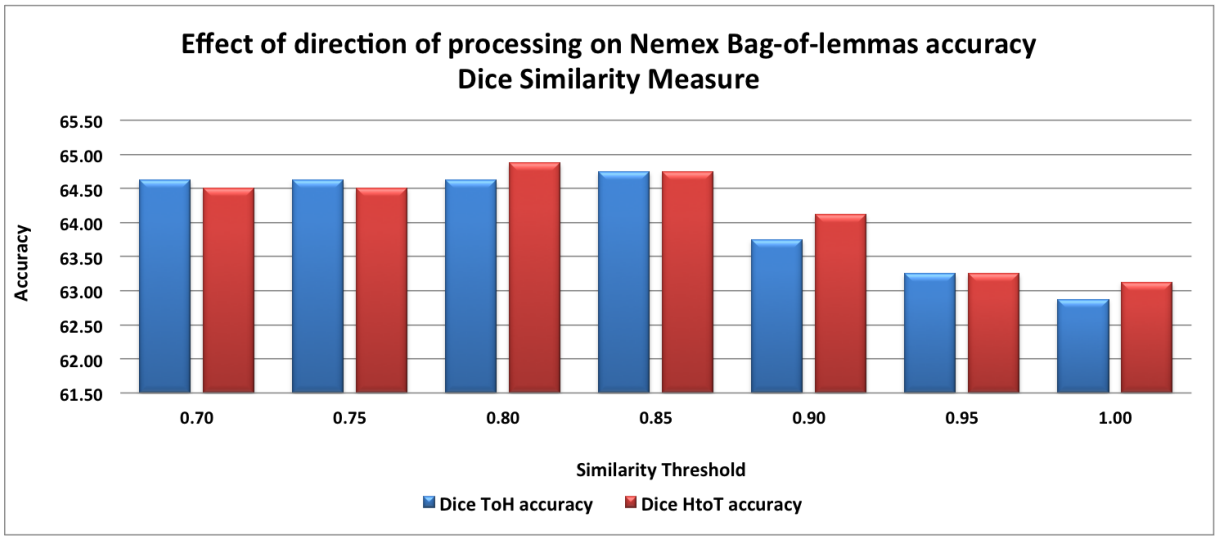


Figure 18. Effect of direction of processing: Nemex bag-of-lemmas scoring, Dice similarity metric

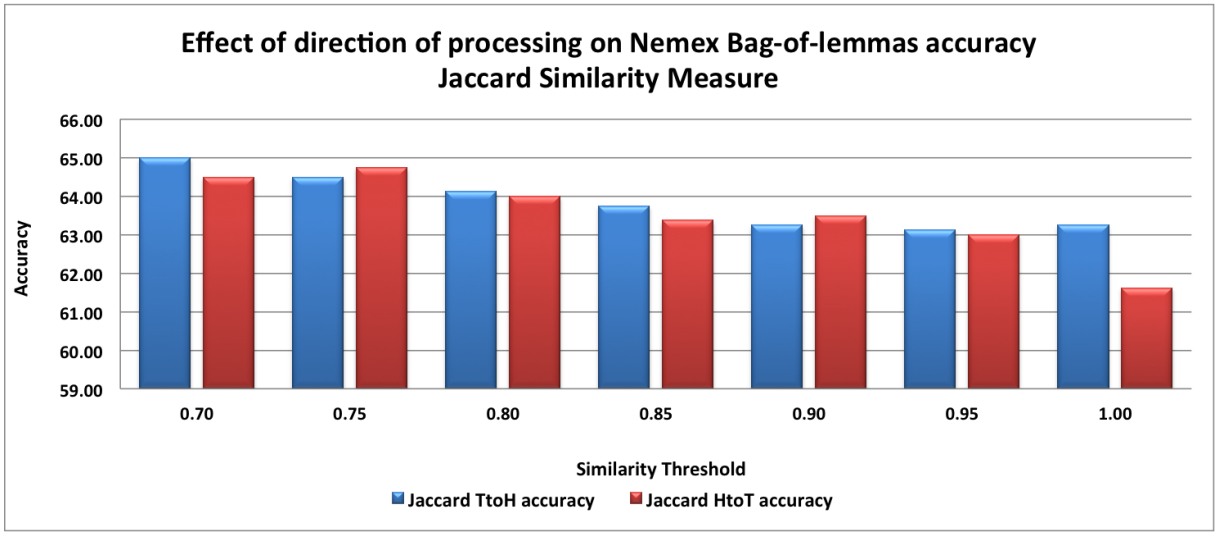


Figure 19. Effect of direction of processing: Nemex bag-of-lemmas scoring, Jaccard similarity metric

Threshold	Dice accuracy	Cosine accuracy	Jaccard accuracy	Dice accuracy with coverage	Cosine accuracy with coverage	Jaccard accuracy with coverage
0.60	58.38	58.00	56.88	59.00	58.13	55.88
0.65	58.00	58.50	55.63	57.63	57.75	57.38
0.70	56.88	56.38	55.88	57.50	56.75	57.00
0.75	56.25	56.38	56.88	55.88	56.13	56.50
0.80	55.88	56.25	56.75	57.13	56.88	56.38
0.85	57.00	57.75	57.25	56.38	56.38	57.25
0.90	57.75	56.75	58.00	56.75	56.75	56.63
0.95	58.25	57.88	58.00	56.50	56.63	56.75
1.00	57.88	58.38	57.63	56.75	57.13	57.38

Table 8. Identifying the best similarity measure and threshold: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T

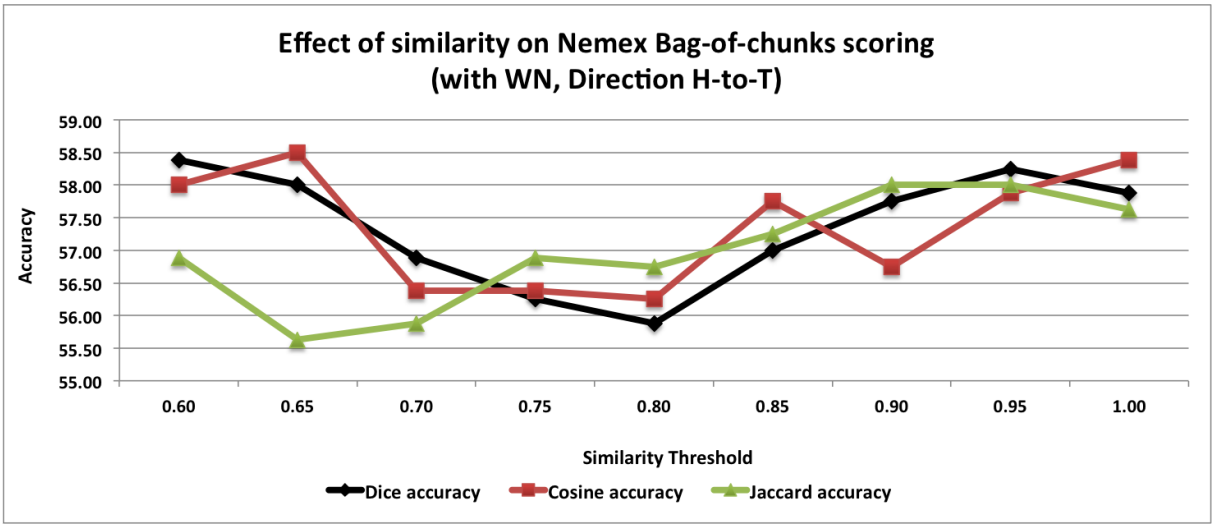


Figure 20. Identifying the best similarity measure and threshold: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T

alignments that reduce classification accuracy, instead of adding meaningful semantic information.

After analyzing the performance of Nemex bag-of-lemmas scorer, experiments were conducted to identify the best configuration for Nemex bag-of-chunks scorer. Chunk entries were expanded using matching WordNet entries. Direction of processing was set as "H-to-T" to limit the number of chunk entries after WordNet expansion. Table 8 lists the accuracy scores for this configuration under different similarity measures and thresholds. A visualization of these results in presented in Figures 20 and 21.

It is observed that accuracy is poor for thresholds in the range 0.7-0.8 for all the similarity measures. Jaccard similarity measure performs poorly as compared to Dice and cosine similarity measures when similarity threshold is low.

When alignments between phrases are identified, phrases with high lexical dissimilarity,

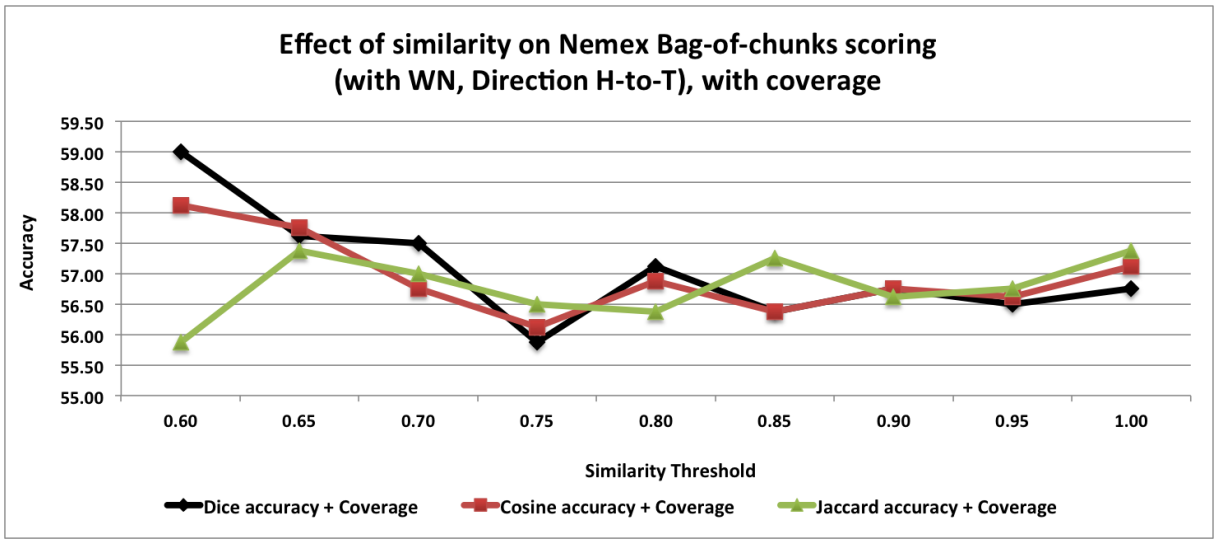


Figure 21. Identifying the best similarity measure and threshold: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T, with coverage features

but also high semantic similarity should be aligned. For example, an alignment should be generated between "the cat and the dog" and "the pet". Unless similarity threshold is sufficiently low, Nemex bag-of-chunks scorer can not align these instances. It is believed that threshold range 0.7-0.8 is sufficiently low to add noise, but not to capture such lexical differences. A lower threshold is required for a more semantic alignment. A higher threshold generates alignments only among almost exact phrases, thereby resulting into a better accuracy.

Further, an effect of coverage features has been analyzed in Figures 22, 23 and 24. Coverage features typically reduce classification accuracy when phrases are aligned through NemexA.

Following the evaluations with Nemex bag-of-chunks scorer, system performance using the semantic phrase scorer was evaluated. Threshold for cosine similarity between vectors for phrases was varied to identify the best configuration. Corresponding accuracy obtained on the RTE-3 data-set is recorded in Table 9. Further, these results have been represented in Figure 25. An analysis of utility of coverage features has also been performed.

Unlike effect of coverage features in Nemex bag-of-chunks scorer, here it is observed that better accuracy is obtained on higher thresholds when coverage features are also used to analyze system performance. However, on lower thresholds, the system performs better when coverage features are not used. Further, it is also observed that semantic phrase scoring results in better accuracy than Nemex bag-of-chunks scoring. It suggests that better alignments are obtained when phrases are semantically aligned using word embeddings, rather than through WordNet extension.

Once the best threshold and feature set for semantic phrase scorer has been identified,

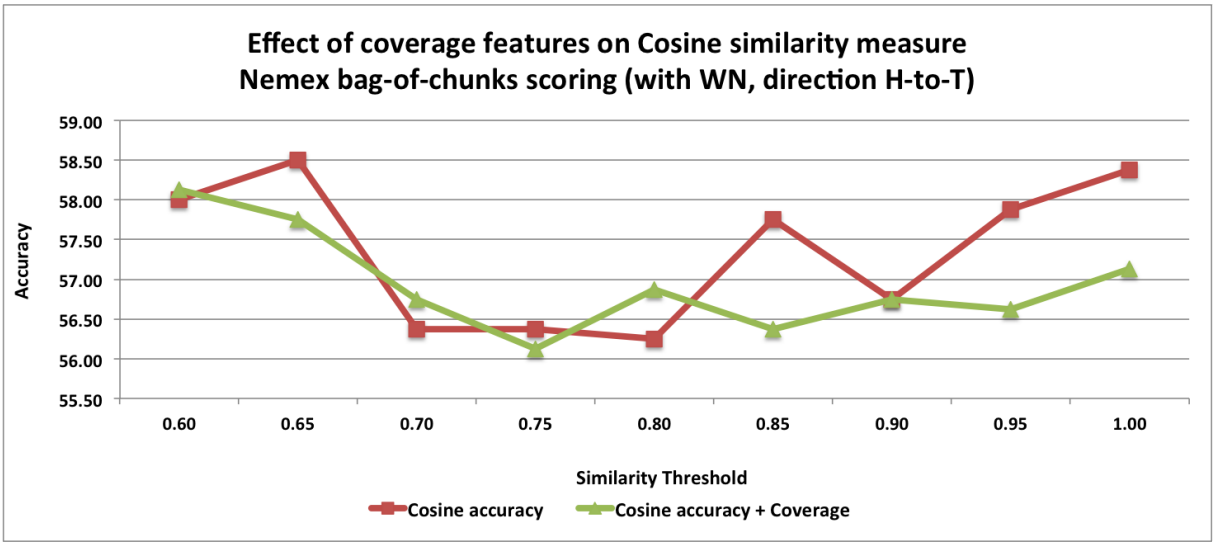


Figure 22. Effect of coverage features on Cosine similarity measure: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T

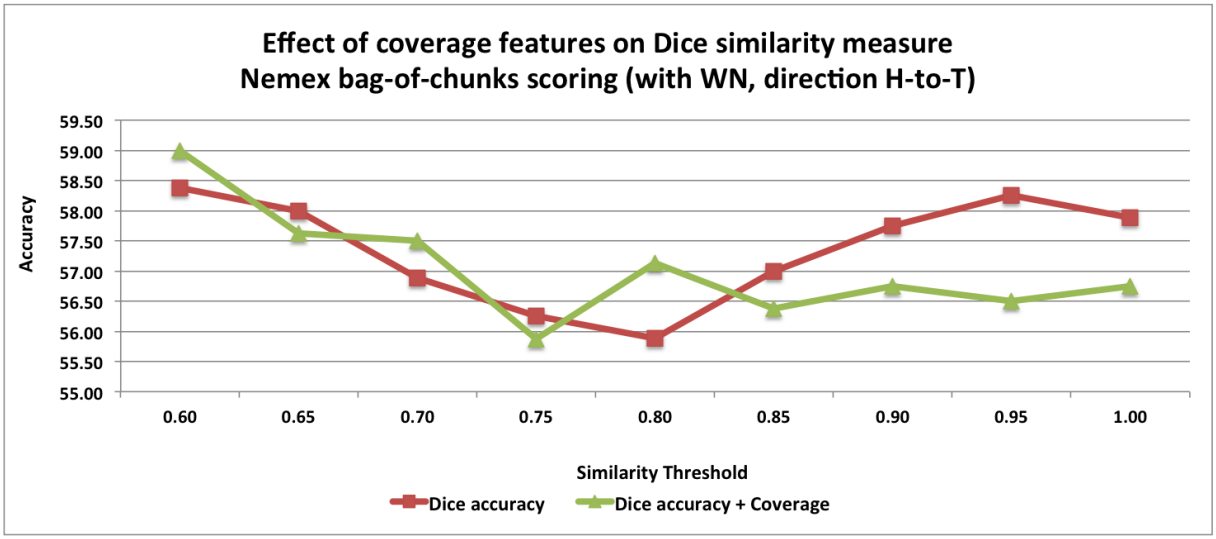


Figure 23. Effect of coverage features on Dice similarity measure: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T

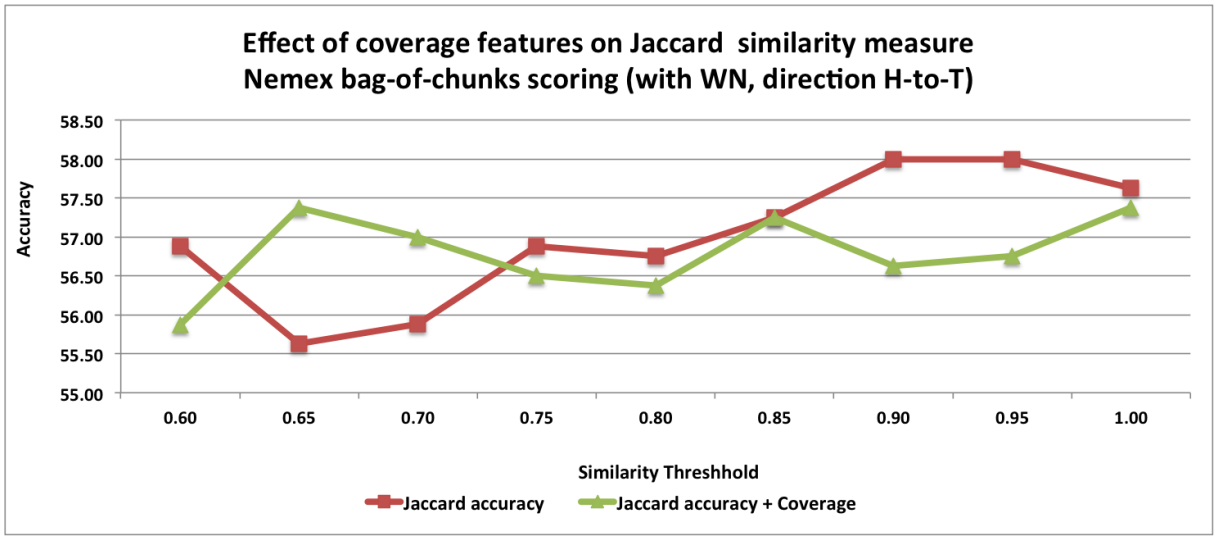


Figure 24. Effect of coverage features on Jaccard similarity measure: Nemex bag-of-chunks Scoring supplemented with WordNet, direction H-to-T

Threshold	Accuracy Without Coverage	Accuracy With Coverage
0.60	59.38	57.75
0.65	59.00	58.50
0.70	58.38	58.38
0.75	57.75	57.13
0.80	56.50	58.38
0.85	56.00	57.88
0.90	54.63	56.13
0.95	55.75	55.88
1.00	55.63	56.63

Table 9. Accuracy using Semantic Phrase Scorer, with and without coverage features, without negative alignment identification

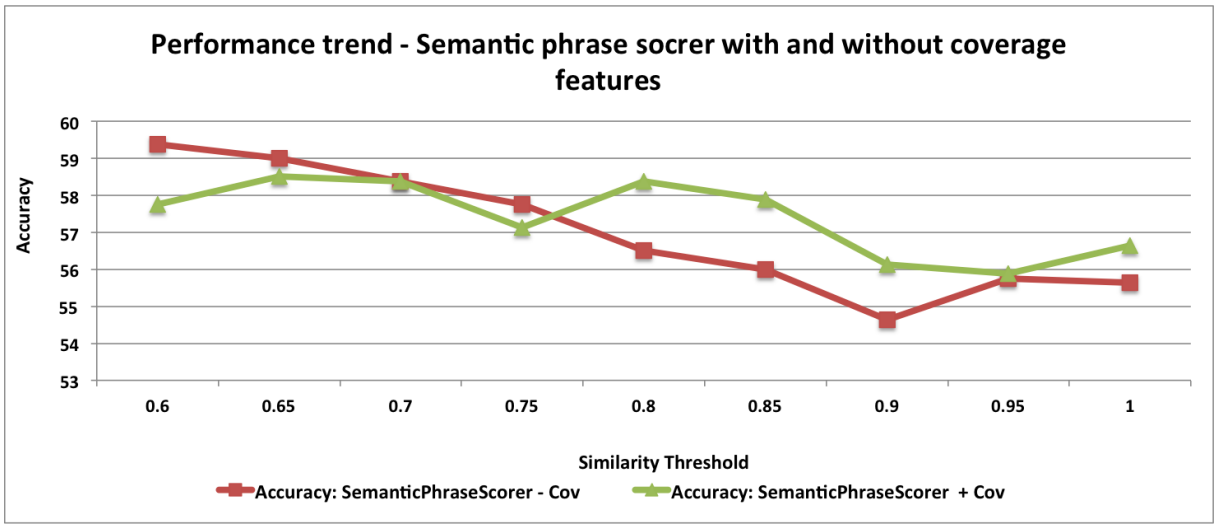


Figure 25. Performance trend - Semantic Phrase Scorer, with and without coverage features, without negative alignment identification

Threshold	WN/VO	Accuracy	Accuracy Without WN,VO
0.6	WN	59.63	59.38
0.6	VO	59.50	59.38
0.6	WN, VO	59.75	59.38
0.9	WN	54.63	54.63
0.9	VO	54.63	54.63
0.9	WN, VO	54.63	54.63

Table 10. Effect of negative alignment identification, Semantic Phrase Scoring without coverage features

an effect of distinction between negative and positive alignment links for entailment has been evaluated. Various experiments are conducted to identify this effect, and the corresponding results obtained are presented in Table 10.

It is observed that distinguishing between negative and positive alignment links improves system performance on lower thresholds, however no change is seen on higher thresholds. Usage of both WordNet (WN) and VerbOcean (VO) for this purpose positively contributes to system performance.

This behavior is witnessed because when similarity threshold is high, majority of the generated alignment links are synonymous. However, reduction in threshold results in addition of links such that context of usage is the same, however inherent meanings are different. Proposed mechanism for negative links identification manages to capture such links that are added in the data.

Threshold	Accuracy With Negation	Accuracy Without Negation
0.6	59.63	59.38

Table 11. Effect of negation scoring on semantic phrase scoring

Nemex BoW Config	Nemex BoL Config	Nemex BoChunks Config	Semantic Phrase Scoring Config	Negation Scoring	Accuracy
Cosine 0.8 T-to-H no stopwords	NA	NA	NA	No	64.88
Cosine 0.8 T-to-H no stopwords	NA	Cosine 0.65 H-to-T WN	0.6 coverage features absent WN VO	Yes	63.75
Cosine 0.8 T-to-H no stopwords	NA	NA	0.6 coverage features absent WN VO	Yes	**64.63**
Cosine 0.8 T-to-H no stopwords	NA	NA	0.6 coverage - content, verb, proper noun WN VO	Yes	63.50
Cosine 0.8 T-to-H no stopwords	Dice 0.75 T-to-H WN	NA	0.6 coverage - content, verb, proper noun WN VO	Yes	63.25

Table 12. Best configurations on RTE-3 data-set

Lastly, effect of negation scoring on semantic phrase scoring has been evaluated in a configuration such that similarity threshold is 0.6, and negative alignments are not distinguished from positive ones. Corresponding result has been recorded in Table 11. Negation scoring has been found to improve the system performance. Although an improvement in accuracy is not very high in terms of absolute numbers, it is significant nonetheless. Its significance is high due to the fact that in an small data-set, the number of instances where such a feature is present is low.

Further, in the end, different available scorers have been combined, and their amalgamated performance is recorded in Table 12. Thereby, the best obtained accuracy is compared with the existing state-of-the-art in Table 13.

It is observed that the best obtained system performance is good, but not better than the best performing systems on RTE-3 data. The developed system achieved an accuracy

System	Accuracy
Proposed best	**64.63**
Textual Inference Engine (TIE) best	65.20
Multilevel Alignment Architecture	67.00

Table 13. Comparison of accuracy obtained on RTE-3 data-set with state-of-the-art

of 64.63% when semantic alignments where added along with approximate bag-of-words scoring. This accuracy is a quarter percent lower than the accuracy obtained with approximate bag-of-words scoring configuration without any semantic information. However, the best available accuracy of Textual Inference Engine, the system used as a basis for this thesis, is 65.20%. Similarly, accuracy obtained using the Multilevel alignment architecture on this data-set is 67.00%. These systems have been described in Sections 1.3.4 and 1.3.3 respectively.

This behavior suggests that alignment and entailment classification techniques besides the one proposed are better for RTE-3 data. However, it is also believed that the RTE-3 data-set is very small and is not an effective indicator for performance of statistical systems. Further evaluations on the RTE-6 and SNLI data-sets has been performed to assess this hypothesis.

3.2.2 Evaluations - RTE-6 data-set

Pairwise evaluations on the RTE-6 data-set have been performed after converting it to the format of RTE-3 data-set. A data-set consisting of (T,H) pairs has been generated by associating each candidate text sentence with its corresponding hypothesis. Generated training set consists of all the candidate sentences and hypotheses from all the topics in the original training set. However, a separate test data-set is created for each topic in the RTE-6 test set, in format of RTE-3 test-set.

During evaluations, separate precision, recall and F-score is recorded for each topic in the test set. These scores are thereby averaged to generate micro and macro-averaged precision, recall and F-score.

Given that this data-set contains 95% negative cases, to achieve a balance between precision and recall of positive cases, misclassification costs are taken into consideration. A cost matrix is used to learn a cost-sensitive model. Training instances are re-weighted according to costs specified in the cost matrix before a model is learned.

Further in this section, evaluation result has been discussed. First of all, it is discussed

Cost ratio entailment:non-entailment	Precision	Recall	F-score
1.0	68.1	7.1	12.9
1.5	59.4	12.7	20.9
2.0	53.2	17.6	26.5
2.5	50.1	23.3	31.8
3.0	44.4	28.2	34.5
3.5	40.3	31.3	35.2
4.0	38.6	34.4	36.4
4.5	36.7	38.2	37.5
5.0	35.2	41.1	37.9
5.5	33.7	43.0	37.8
6.0	32.4	46.4	38.1
6.5	30.9	49.1	37.9
7.0	30.1	51.1	38.0
7.5	29.1	53.8	37.8
8.0	28.7	56.4	38.0
8.5	27.9	58.3	37.8
9.0	27.1	59.9	37.3
9.5	26.1	60.5	36.5
10.0	25.4	61.9	36.0

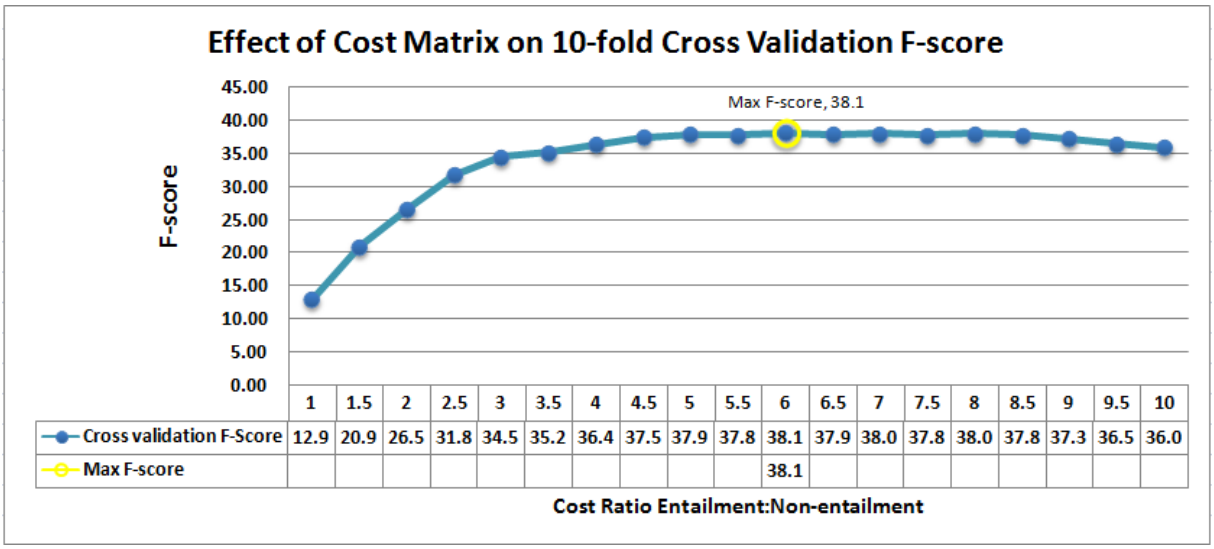
Table 14. Tuning cost ratio, entailment:non-entailment using 10-fold cross-validation on RTE-6 development set

how a cost matrix is decided upon, and then the obtained F-score is compared with the state-of-the-art results.

Multiple configuration parameters of the system are assumed to follow the same trend for this data-set, as in the RTE-3 data-set. The utilized configuration is Nemex bag-of-words scoring with cosine similarity measure and similarity threshold of 0.8, semantic phrase scoring with threshold 0.75, using WordNet to find negative links, along with content word, nouns and proper nouns coverage features, and negation scoring. Different cost ratios have been considered and the optimal one is chosen based on cross-validation approach on training data.

Using the previously specified configuration, F-scores obtained for 10-fold cross-validation on the positive class in the development set is given in Table 14. Figure 26 presents these results in a graph and indicates the ratio corresponding to the maximum F-scores.

From Table 14, the cost ratio corresponding to maximum F-score, which weighs the



	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Cross validation F-Score	12.9	20.9	26.5	31.8	34.5	35.2	36.4	37.5	37.9	37.8	38.1	37.9	38.0	37.8	38.0	37.8	37.3	36.5	36.0
Max F-score											38.1								

Figure 26. Tuning cost ratio, entailment:non-entailment using 10-fold cross-validation on RTE-6 development set

positive instances 6 times the negative ones, is chosen as the standard cost ratio for evaluating performance further on the test set. Obtained precision, recall and F-score for each topic in the test set, according to previously chosen best configuration, has been input in Table 15.

Test Set Topic ID	Precision	Recall	F-score
D_0901	26.97	68.91	38.77
D_0902	26.62	44.09	33.20
D_0907	28.09	67.57	39.68
D_0913	25.41	56.10	34.98
D_0918	37.50	60.64	46.34
D_0928	43.81	46.46	45.10
D_0931	34.36	72.90	46.71
D_0936	42.86	57.35	49.06
D_0939	25.00	46.09	32.42
D_0943	41.53	78.63	54.35
Micro-average	32.33	60.32	42.10
Macro-average	33.22	59.87	42.73

Table 15. Precision, recall and F-score for each topic RTE-6 test-set using specified configuration

A micro-average F-score of 42.10 is obtained on the RTE-6 data-set when parameters are chosen in such a manner. This score is compared with the state-of-the-art systems in

System	Micro-average F-score	Macro-average F-score
Baseline (best)	34.63	NA
Proposed best	**42.10**	**42.73**
LITE	39.81	40.58
RTE-6 challenge best	48.01	49.58

Table 16. Comparison of accuracy obtained on RTE-6 data-set with state-of-the-art

Table 16.

It is observed that the proposed system is among the top performing systems in the RTE-6 challenge, although it was not tailored only for this data-set. The system performs better than LibLinear Textual Entailment Engine (LITE) developed at DFKI, discussed in Section 1.3.5 earlier. LITE uses Meteor to generate alignment scores between 'T' and 'H', along with several other features. This indicates that the generated alignments are comparable to those generated by Meteor, and the newly proposed features perform better.

3.2.3 Evaluations - SNLI corpus

Lastly, system performance has been evaluated on the SNLI corpus. Two-way entailment relation classification is performed such that "Neutral" and "contradiction" classes are both considered to be "non-entailing" (T,H) pairs. The SNLI corpus is converted to format of the RTE3 data-set, and pairwise processing is thereby done. The pairs in which inter-annotator agreement could not be reached are not considered for processing. This leaves a total of 549,367 pairs in the training set, 9842 pairs in the development set and 9824 pairs in the test set.

Among the available (T,H) pairs in the SNLI corpus, a very small subset of pairs could not be tokenized due to technical challenges. These pairs, numbering 431 in training set, 6 in development set, and 12 in test set, have been disregarded during evaluations.

Evaluations on the SNLI corpus has been performed using the configuration - Nemex bag-of-words scoring with cosine similarity threshold 0.8, Semantic phrase scoring with threshold 0.75, identifying negation links through WordNet, and a Negation scoring component. Task features have not been used during classification, because this information is not associated with the data-set. Performance with and without coverage features, both, have been recorded. A slightly improved performance was obtained when none of the coverage scores were used while learning the model.

System	Accuracy (2-way)
Proposed (without coverage scores)	**75.27**
Proposed (With coverage scores: content word, verb, proper nouns)	75.14
Edit-distance (FBK)	71.90
TIE without lexical resources	72.20
TIE with lexical resources	75.00

Table 17. System performance on two-way classifications on SNLI corpus, and comparison with state-of-the-art

Rows 3 to 5 of table 17 indicate performance of the algorithms within the EOP on the SNLI corpus for a two-way task. Edit-distance based entailment engine has been developed at "Fondazione Bruno Kessler", Italy and uses edit distance operations to decide about entailment. Further, TIE is the system developed at DFKI, discussed in Section 1.3.4. The proposed system has comparable performance with the existing systems in the EOP on the SNLI corpus.

4 Conclusions and Discussion

In this section, the results obtained on the three data-sets will be compared, and conclusions will be discussed. Further, suggestions for extending and improving the system in future will be put forward. Lastly, a critical analysis of the available data-sets and evaluation metrics will be performed.

4.1 Summary and Conclusions

The results obtained in Section 3.1.1 reflect that entailment accuracy improves with an approximate match, as opposed to an exact match. However, when similarity threshold is reduced below a certain value, the accuracy starts decreasing again due to addition of too much noise.

Moreover, it is observed that entailment accuracy in the Nemex bag-of-lemmas configuration follows a similar trend as Nemex bag-of-words configuration. However, this trend is less steep than the previous one. This configuration performs poorly as compared to Nemex bag-of-words configuration. Semantic alignments through WordNet add more noise to the data, than meaningful information.

Furthermore, phrase alignments generated through word embedding result in better system performance than phrase alignments generated using NemexA. Identification of negative alignments through WordNet and VerbOcean while using word embeddings improves performance of the system.

Similarly, accounting for negation terms also positively contributes to entailment classification accuracy.

As we see in Sections 3.2.2 and 3.2.3, accuracy and performance of algorithms improve with an increase in the size of the data-sets. This behavior is in accordance with the behavior expected from statistical systems. Larger number of instances allow learning more generalized models, in our case, for entailment decision classification.

While the small size of the RTE3 data-set is a limiting factor for performance of systems applying machine learning algorithms, larger data-sets like the RTE-6 data-set and the SNLI corpus allow for efficient and more accurate classification using these algorithms. Comparable results have been obtained by the proposed system on these larger data-sets.

4.2 Suggested Improvements

The SNLI corpus was released towards the end of implementation phase of the thesis research. Hence, sufficient time was not available to perform multiple evaluations using this corpus. We believe that an improvement in classification accuracy would be obtained on tuning parameters directly on this corpus, instead of re-using the tuned parameter values for the best available configuration according to the RTE-3 data-set. Therefore, further experimentation with this corpus, which is the new standard for evaluating RTE algorithms, should be performed. Furthermore, 3-way classification should be explored to compare the system performance with the best results on the SNLI corpus.

The proposed system performs lexical and semantic alignments between phrases of text. It adds multiple alignment links between phrases based on exact or approximate text similarity, word stem similarity, semantic similarity through the use of knowledge bases WordNet and VerbOcean, or content level semantic similarity through embedded vectors for phrases. Before computing most of these similarities, phrases have been obtained through the use of a 'chunker'. This 'chunker' identifies phrases in text - noun phrase, verb phrase, or prepositional phrase using a pre-trained maximum entropy classifier. Hence, performance of the applied 'chunker' becomes a bottleneck for system performance. In order to overcome this bottleneck, phrases need to be automatically identified instead.

Moreover, while performing semantic alignment between phrases using word embeddings, vectors for phrases have been generated by combining vectors for constituent words using 'addition' operator. These word vectors have been trained using Word2Vec on the Google News corpus. Although this generates reasonable alignments, the introduction of the SNLI corpus allows for training phrase vectors. The next steps would include learning these vectors from the corpus directly.

Furthermore, the proposed approach adds several lexical and semantic phrase alignment links to analyze whether an entailment relation holds. These semantic links allow aligning phrases that have similar meaning. However, in order to identify whether the meaning of hypothesis is covered by meaning of corresponding text, these links can be enriched with semantic roles. A high correspondence between semantic roles in text and hypothesis would support an entailment judgement, as compared to the opposite.

Finally, negation scores calculated in the developed system do not take into account the scope of negation words. For the negation feature to be more effective, these scopes should be accounted for. Instead of calculating negation scores dependent on the occurrence of negation words in text and hypothesis, relative to their sizes, only those negation words should be considered, which modify some aligned phrase.

In addition to these suggested improvements for a more efficient alignment and feature value generation, discourse resolution should be performed on (T,H) pairs in the RTE-6

data-set. Difference between the developed system for the RTE-3 and the RTE-6 data-set lies in the classifier being used. The same processing is done for the two to calculate feature scores, although the data-sets are very different to each other. Due to the corpus-oriented approach followed by the RTE-6 data-set, several references to such information is present, which is outside the scope of the sentence pair in question. Efficient processing of such references requires discourse resolution prior to generating alignments between phrases of text and hypothesis.

Such enhancements are expected to improve system performance, thereby generating efficient entailment classification models.

4.3 Criticism

Following the conclusions in Section 4.1, it is discernible that the RTE-3 data-set is too small for efficient learning of statistical models. Increase in classification accuracy obtained on moving from the RTE-3 data-set to the SNLI corpus establishes the SNLI corpus as the new standard. However, it does not include (T,H) pairs where text may range up to a paragraph in length, compared to single sentence hypothesis. An ideal corpus for an entailment classification task would be a very large corpus which indicates a more natural entailment distribution, as opposed to artificially constructed cases for entailment.

Furthermore, the Excitement Open Platform (EOP) has been developed with a perspective for pair-wise processing. All the (T,H) pairs in the data-set are annotated as an independent data point. However, data-sets like the RTE-6 data-set follow a corpus-oriented approach, where each (T,H) pair is not independent of other sentences in the corpus. Pair-wise perspective highly inefficient for such a data-set. Instead of reconstructing the RTE-6 data-set in the format of the RTE-3 data-set by enlisting all possible pairs of hypotheses and corresponding candidate sentences, a tool which directly identifies entailing sentences from a corpus given a hypothesis is more practical. It can be directly applied to several tasks like question answering. Hence, abstracting away from obligatory pair-wise processing in the EOP would be recommended for a more pragmatic approach.

The RTE-6 data-set reflects natural distribution of entailment relations in a corpus. It is comparatively larger than the RTE-3 data-set. However, evaluations on this data-set processes each topic independently, and thereby results on these topics are averaged. Each topic consists of a much smaller number of (T,H) pairs, where about 95% of cases are negative. In order to handle this imbalance, a cost matrix has been used. However, if topics were not processed independently, the data-set would be large enough to apply more advanced techniques to handle this imbalance, like ensemble learning.

References

Arasu, A., Ganti, V., and Kaushik, R. (2006). Efficient exact set-similarity joins. In *Proceedings of the 32nd international conference on Very large data bases*, pages 918–929. VLDB Endowment.

Artale, A., Magnini, B., and Strapparava, C. (1997). Wordnet for italian and its use for lexical discrimination. In *AI* IA 97: Advances in Artificial Intelligence*, pages 346–356. Springer.

Baldridge, J. (2005). The opennlp project. *URL: http://opennlp. apache. org/index. html,(accessed 2 February 2012)*.

Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., and Szpektor, I. (2006). The second pascal recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, volume 6, pages 6–4.

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

Bentivogli, L., Clark, P., Dagan, I., Dang, H. T., and Giampiccolo, D. (2010). The sixth PASCAL recognizing textual entailment challenge. In *The Text Analysis Conference (TAC 2010)*.

Bentivogli, L., Dagan, I., Dang, H. T., Giampiccolo, D., and Magnini, B. (2009). The fifth pascal recognizing textual entailment challenge. *Proceedings of TAC*, 9:14–24.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Chaudhuri, S., Ganti, V., and Kaushik, R. (2006). A primitive operator for similarity joins in data cleaning. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 5–5. IEEE.

Chklovski, T. and Pantel, P. (2004). Verbocean: Mining the web for fine-grained semantic verb relations. In *EMNLP*, volume 2004, pages 33–40.

Clinchant, S., Goutte, C., and Gaussier, E. (2006). Lexical entailment for information retrieval. In *Advances in Information Retrieval*, pages 217–228. Springer.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Dagan, I., Glickman, O., and Magnini, B. (2006). The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Dagan, I., Roth, D., and Sammons, M. (2013). *Recognizing textual entailment: models and applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, San Rafael.

de Castilho, R. E. and Gurevych, I. (2014). A broad-coverage collection of portable nlp components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT) at COLING*, pages 1–11.

Denkowski, M. and Lavie, A. (2011). Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 85–91. Association for Computational Linguistics.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Ferrucci, D. and Lally, A. (2004). Uima: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3-4):327–348.

Giampiccolo, D., Dang, H. T., Magnini, B., Dagan, I., Cabrio, E., and Dolan, B. (2009). The fourth pascal recognizing textual entailment challenge. In *Proceedings of the First Text Analysis Conference (TAC 2008)*. Citeseer.

Giampiccolo, D., Magnini, B., Dagan, I., and Dolan, B. (2007). The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The Journal of Machine Learning Research*, 13(1):307–361.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hamp, B., Feldweg, H., et al. (1997). Germanet-a lexical-semantic net for german. In *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15. Citeseer.

Harabagiu, S. and Hickl, A. (2006). Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 905–912, Stroudsburg, PA, USA. Association for Computational Linguistics.

Harrell, F. E. (2013). *Regression modeling strategies: with applications to linear models, logistic regression, and survival analysis*. Springer Science & Business Media.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Magnini, B., Zanoli, R., Dagan, I., Eichler, K., Neumann, G., Noh, T.-G., Pado, S., Stern, A., and Levy, O. (2014). The excitement open platform for textual inferences. *ACL 2014*, page 43.

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

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

Mikolov, T., Yih, W.-t., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751.

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

Ng, A. Y. (2004). Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM.

Nivre, J., Hall, J., and Nilsson, J. (2006). Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC*, volume 6, pages 2216–2219.

Noh, T.-G., Padó, S., Shwartz, V., Dagan, I., Nastase, V., Eichler, K., Kotlerman, L., and Adler, M. (2015). Multi-level alignments as an extensible representation basis for

textual entailment algorithms. *Lexical and Computational Semantics (* SEM 2015)*, page 193.

Okazaki, N. and Tsujii, J. (2010). Simple and efficient algorithm for approximate dictionary matching. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 851–859. Association for Computational Linguistics.

Ostermann, S., Horbach, A., and Pinkal, M. (2015). Annotating entailment relations for shortanswer questions. *ACL-IJCNLP 2015*, page 49.

Padó, S., Noh, T.-G., Stern, A., Wang, R., and Zanoli, R. (2014). Design and realization of a modular architecture for textual entailment. *Journal of Natural Language Engineering*.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the international conference on new methods in language processing*, volume 12, pages 44–49. Citeseer.

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

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85.

Volokh, A. and Neumann, G. (2011). Using mt-based metrics for rte. In *The Fourth Text Analysis Conference. NIST*.

Wang, R. (2011). *Intrinsic and extrinsic approaches to recognizing textual entailment*. PhD thesis, Saarland University.

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

Zeller, B. D., Snajder, J., and Padó, S. (2013). Derivbase: Inducing and evaluating a derivational morphology resource for german. In *ACL (1)*, pages 1201–1211.