This is a draft version of a paper submitted for a journal

A Lexico-Syntactic-Semantic approach for Recognizing Textual Entailment

Rohini Basak[1], Sudip Kumar Naskar[1], Alexander Gelbukh[2]

[1] Jadavpur University, Kolkata,
India

[2] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City,
Mexico

visitrohinihere@gmail.com, sudip.naskar@cse.jdvu.ac.in, gelbukh@gelbukh.com

Abstract. This paper presents a rule-based technique for recognizing textual entailment. The method exploits lexical similarity, semantic similarity and dependency relations for this purpose. First, dependency parsing is applied on a text–hypothesis (T–H) pair to produce a directed dependency graph each for the text and the hypothesis. The proposed method compares the hypothesis dependency graph with the corresponding text dependency graph to check whether the hypothesis can be inferred from the text. For each dependency triplet of the hypothesis, the method tries to find out a matching dependency triplet of the text. This matching is carried out on the basis of a number of matching rules which have been designed after a thorough and detailed analysis of the RTE development sets. Upon finding a successful match for a hypothesis triplet $R^H(W_1,W_2)$ with any of the text triplets, a matching score in the range of [-1, +1] is assigned to the corresponding dependent node (W_2) associated with that particular arc in the hypothesis graph. The match score is computed based on the lexical or semantic similarity between the corresponding text–hypothesis governor words or the dependent words associated with the matching pair of dependency triplets. If for any hypothesis dependency triplet, no corresponding matching triplet is found in the text, a 0 score is assigned to W_2. Finally the hypothesis dependency graph is traversed in level order fashion starting from the bottommost level from where the scores of the nodes are accumulated and propagated to the upper levels. This traversal process stops at the topmost level where the final entailment score between the T–H pair is obtained at the root node. This similarity score is then compared against a predetermined threshold value to take the final entailment decision. The threshold values were tuned on the task specific splits of the RTE development sets using a hill climbing approach. Experimental results on the RTE datasets show 60.5%, 64.4%, 62.8% and 61.5% accuracy for the RTE1, RTE2, RTE3 and RTE4 testsets respectively on the 2–way entailment task. The method attains an accuracy of 54.3% for the 3–way TE task on the RTE4 testset.

Keywords: Textual entailment, Dependency parsing, lexical matching, semantic similarity, dependency relation matching, rules, RTE datasets.

1. Introduction

Recognizing Textual Entailment (RTE) can be defined as the task of taking a pair of text fragments as input and deciding whether the meaning of one fragment (termed as *hypothesis*) can be inferred from the meaning of the other fragment (termed as *text*) or not. The task of RTE has attracted the attention of many researchers in the natural language processing (NLP) community over the last few decades as it has important applications in many NLP tasks ranging from Information Extraction (IE), Information Retrieval (IR), Question Answering

(QA), Summarization (SUM), Paraphrase Acquisition (PP), Machine Translation (MT), Reading Comprehension (RC) etc. Due to the huge impact in many application areas, PASCAL (Pattern Analysis, Statistical Modelling and Computational Learning) has organized a number of RTE challenges over the last few years.

Textual Entailment is a directional relationship between a pair of text (T) and hypothesis (H). If T textually entails H, then the meaning of H can be inferred from T, however, the reverse may not be true always. Let us consider the following example T–H pair taken from the RTE1 dataset.

Example 1:

<pair id="122" value="TRUE" task="IR">
<t>The Daily Telegraph, most prized asset in Lord Conrad Black's crumbling media empire, has been sold to Britain's Barclay twins.</t>
<h>Daily telegraph is sold.</h>
</pair>

From the T–H pair shown in Example 1, it is clear that the meaning of the hypothesis (contained in <h></h> in the example) can easily be inferred from that of the text (contained in <t></t> in the example). However, since the text contains some additional information other than those contained in H, therefore, one cannot say that the truth of T can be derived from that of H. Therefore, TE is a unidirectional relation that holds only from T to H, but not in the reverse direction. Textual entailment is a 2 way classification task which means that a given T–H pair may be labelled either as a case of YES entailment or NO entailment. This 2 way entailment task has been later extended to 3 way task in the 4th RTE challenge organized by PASCAL. The 3 way RTE task marks a given T–H pair with one of three possible labels - ENTAILMENT, CONTRADICTION and UNKNOWN. The following example taken from RTE4 dataset depicts a case of CONTRADICTION.

Example 2:

<pairid="45"entailment="CONTRADICTION" task="IR">
<t>As German voters go to the polls on Sunday, unemployment will be a key issue. Despite tough labour market reforms, the number of unemployed has risen to 5m. And Germany's jobless are getting despondent.</t>
<h>Germany's jobless rate decreases.</h>
</pair>

A human reading of the T–H pair in Example 2 reveals that the fact conveyed by the hypothesis totally opposes or contradicts the information stated in the text. Therefore, it should be marked as a case of CONTRADICTION rather straightforwardly.

Another example T–H pair from the RTE4 dataset is presented in Example 3 which indicates a case of UNKNOWN.

Example 3:

<pairid="61"entailment="UNKNOWN" task="IR">
<t>Fifty one persons were killed today when two trains travelling on the same track collided in northern Egypt in the country's deadliest rail crash in four years. Two carriages were derailed in a tangle of torn metal as one train slammed into the back of another.</t>
<h>A French train crash killed children.</h>

The truth of the hypothesis of Example 3 cannot be inferred from the given text. The text simply conveys that *fifty one persons were killed* from which we cannot conclude whether any *children were killed* or not. Therefore, this is evident that this T–H pair depicts an UNKNOWN case for TE due to the absence of relevant or supporting information in the text to verify the truth of the hypothesis.

In this paper, we presented a rule-based approach to recognizing textual entailment which combines lexical, syntactic and as well as semantic features. First the T–H pair is subjected to a number of preprocessing operations and then it is passed to a dependency parsing module. This module produces a set of dependency triplets from which a dependency graph is constructed separately for each of the text and hypothesis. The method next tries to align the hypothesis tokens with the text tokens on the basis of a number of WordNet based lexical relations. After the text and the hypothesis tokens have been aligned, each dependency triplet of the hypothesis is searched against all the text dependency triplets in order to establish its corresponding matching triplet. This matching is carried out on the basis of a number of dependency triplet matching rules which exhibit mainly syntactic divergences between the lexically aligned tokens. On finding a successful matching text dependency triplet, a matching score is assigned to the child node of that branch of the hypothesis dependency graph which constitutes the particular dependency triplet. These matching rules were synthesized after a detailed and minute study of the various RTE development sets. A set of equivalent relation pairs were identified by analysing the development sets of the RTE datasets. These equivalent relation pairs are largely universal and can be applied to any dataset for the task of recognizing textual entailment. If for any hypothesis dependency triplet, one of the tokens is not aligned with any of the text tokens, the semantic similarity module is invoked to align the unaligned hypothesis token with semantically the most similar token in the text. Finally the hypothesis dependency graph is traversed in level order and the scores of all the nodes are calculated in a bottom up fashion by accumulating the scores of the nodes belonging to a particular level and then propagating it to the upper level. The traversal continues till the final entailment score between the T–H pair is obtained at the root node at the topmost level of the graph. This score is then compared with a predetermined threshold value above which a YES entailment decision is taken, otherwise a NO entailment decision is made. The threshold values are learnt from the development sets. Our method is also able to correctly label a good number of T–H pairs of the RTE4 dataset for the 3 class TE problem. The rest of the paper is organized as follows: Section 2 reviews works related to RTE. Section 3 presents the modularized architecture of our method. The details of the datasets on which we performed our experiments are discussed in section 4. The experiments and the results are reported in section 5. Section 6 provides an analysis of the degree of effectiveness of the synthesized matching rules in overall performance and also an in depth error analysis. Section 7 concludes and provides avenues for future works.

2. Related Work

Over the years, several methods have been proposed to solve the problem of recognizing textual entailment. Some of them represent the text and hypothesis as syntactic or dependency parse trees and take the entailment decision based on how much of the hypothesis dependency tree is included in the text tree. Many systems simply use some form of lexical matching such as n-gram matching, percentage of word overlap, finding longest common subsequence, skip gram matching, etc. Recently machine learning based classification algorithms have been used extensively by many researchers for the textual entailment task. Many systems also use some form of semantic techniques such as semantic role labelling, atomic propositions, inference rules, universal networking language augmented with semantic similarity measures, etc.

Here we have mainly focused on the works based on tree structure of a sentence using syntactic or dependency tree representation.

Herrera et al. (2006) parses the T–H pair using Lin's Minipar and then finds a lexical entailment between each word in the hypothesis with some word in the text on the basis of several WordNet relations such as synonym, hyponym, antonym etc. The system then checks whether the dependency tree of the hypothesis is completely or partially included in the text tree to conclude how semantically similar are the two text snippets. The degree of inclusion of the hypothesis tree into the text tree determines the entailment decision.

The main idea behind the work described in (Rios and Gelbukh, 2012; Kouylekov and Magnini, 2005) is based on the assumption that a given T–H pair holds an entailment relation if a finite sequence of edit operations such as insertion, deletion or substitution can be performed on T to produce H with an overall cost below a certain threshold.

Marsi et al. (2006) adopted a dynamic programming approach to establish the concept of normalized alignment of the text and hypothesis dependency trees for predicting the entailment decision. The tree alignment algorithm calculates a matching score between each node in the hypothesis dependency tree with each node in the text dependency tree. However the matching score between any pair of nodes does not depend on the similarity of those nodes only, rather it is computed recursively based on the scores of the best matching pairs of their descendents.

The main motivation behind the work presented in (Snow et al., 2006) is to recognize false entailment. The textual entailment recognition system parses the given T–H pair by NLPwin parser and then represent them as graphs of syntactic dependencies. Each node in H is attempted to be aligned with a node in T using a set of syntactic heuristics such as exact and synonym match, numeric value match, acronym match etc. Then it is checked whether the alignment between any pair of text–hypothesis node belong to another set of syntactic heuristics such as antonym match, negation mismatch, superlative mismatch etc. On satisfying any of these heuristics, the given T–H pair is predicted as false entailment.

Haghighi et al. (2005) adopted a graph based representation of sentences and used a learned graph matching approach to measure the semantic overlap of text. Each vertex of the hypothesis graph is attempted to be mapped with some text vertex by using exact match, stem match, synonym match, hypernym match etc. The entailment decision between the T–H pair is taken by measuring the number of matching text–hypothesis vertices as well as to what extent the relationships between the vertices of the hypothesis are preserved in their text counterpart. The relation or the path between any two vertices is tried to be matched by exact match, partial match, ancestor match etc. Finally weights are learnt according to the relative importance of the vertex and relationship matches to approximate the amount of semantic content in the hypothesis which is contained in the text.

The aim behind the work in (Blake, 2007) is to explore the degree to which the sentence structure plays a role in detecting textual entailment. Each of the text fragments of a given T–H pair is individually parsed by the Stanford parser to generate a typed dependency tree. After collapsing the preposition paths in the generated trees, the system uses a number of base level sentence features such as subject, object, verb, proposition and some derived sentence features such as subject–subject, verb–verb, subject–verb to record the number of matches and the percentage of matches for each hypothesis sentence supported by the text sentences. The final entailment decision is taken by combining the derived features by using a decision tree learning algorithm.

A dependency parser based textual entailment system is described in (Pakray et al., 2010a). The system extracts syntactic structures from the T–H pair by CCG and Stanford parser separately. Successively, the hypothesis relations are compared with the text relations on the basis of different features like subject, object, noun, verb etc and different weights are assigned for exact and partial matches. Finally, all these weights are summed up and checked against a threshold value to take the final entailment decision.

Pakray et al. (2010b) reported a syntactic textual entailment system that extracts the dependency relations from T and H using Stanford parser. Each of the hypothesis relations is then compared with the text relations to find for a complete match or partial match based on some comparisons such as subject–subject, subject–verb, object–verb comparisons etc. Different weights learnt from the development set are assigned on finding a complete or partial match. An optimal threshold has been set on the fraction of matching hypothesis relations based on the development set and it is then applied on the test set to decide the presence or absence of entailments.

The TE recognition systems in (Pakray et al., 2011a,b) use semantic features based on universal networking language (UNL) to predict the entailment decision. UNL expresses information or knowledge in the form of semantic network made up of a set of binary relations. The T–H pairs are first converted to UNL expression using the UNL En-Converter. Each hypothesis UNL relation is then compared with the text UNL relations on the basis of a number of expanded UNL relations, several rules such as Relation Grouping rule, Named Entity Rule etc and some WordNet relations. Different matching scores are considered upon satisfying the matching rules. The final obtained score above a certain threshold indicates a YES entailment, otherwise a NO entailment decision is taken.

Dinu and Wang (2009) proposed an inference rule based technique for recognizing textual entailment. They start with a collection of inference rules acquired automatically based on the distributional hypothesis and propose methods to refine it and obtain more rules using a hand-crafted lexical resource. A dependency based structure representation from the texts is used to provide a proper base for the inference rule application.

Basak et al. (2015) illustrated a method that takes the entailment decision by comparing the dependency tree structures of T and H based on a number of dependency triplet matching rules. The matching rules are synthesized by minutely analysing the PETE dataset. On finding a matching text triplet for a hypothesis dependency triplet, a matching score of 1 is assigned to the dependent node associated with that hypothesis triplet. Finally the hypothesis dependency tree is traversed in post order fashion to accumulate the score of the nodes belonging to one level and then propagating it to the upper levels. The entailment score between the T–H pair is obtained at the root node of the tree at the end of the traversal process. However, the aim behind this work is just to emphasize the role of dependency parser alone in the task of RTE. Therefore no other knowledge sources or NLP tools are embedded in the system. The motivation behind our present work is largely influenced by this particular work and we present a significantly extended version of this method augmented with lexical resources, semantic similarity measures and a large set of matching rules.

3. The Method

This section describes our proposed method. Figure. 1 shows the schematic diagram of the method. The components of the method are illustrated in the following subsections.

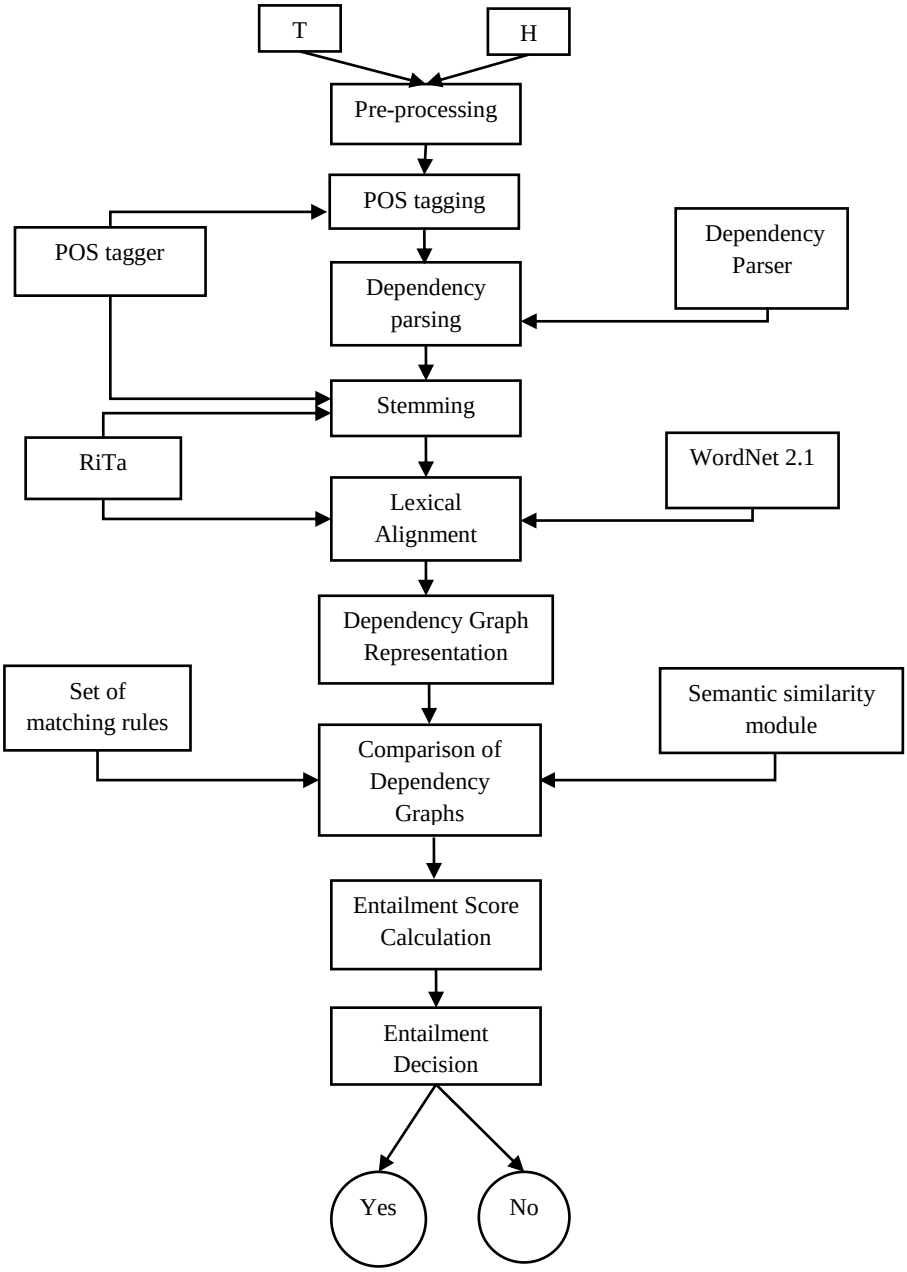

Figure. 1. Modularized System Architecture

3.1 Pre-processing

Before the actual textual entailment detection process is invoked, the T−H pairs are subjected to various pre-processing operations T–H which are described below.

3.1.1 Breaking of Hyphenated-words

There are several T–H pairs in the RTE datasets where either T or H contains one or more hyphenated words while the other does not, which creates problems in lexically aligning the hypothesis tokens to the text tokens. This step breaks such hyphenated words into their component words by removing the hyphens.

3.1.2 Normalization

This step involves normalization of numeric expressions which replaces the occurrence of comma from a numeric value (e.g., 8,568 → 8568).

3.1.3 Replacement of some special symbols

This pre-processing step replaces the occurrences of some special symbols with their corresponding expanded forms. A few of such special symbols and their corresponding expanded forms are presented in Table 1.

Table 1: Special symbols and their expanded forms

Symbol	Expanded form
$	Dollar
£	Pound
%	Percent
m	million/meter
bn	Billion
km	Kilometre

A symbol is replaced in the text (or hypothesis) if its corresponding expanded form is present in the hypothesis (or text). Otherwise, the symbol is left unaltered. Replacement of the symbols based on the presence of their expanded forms in the T–H pair counterpart also makes the ambiguous case of 'm' (meter or million) trivial.

3.1.4 Expansion of contracted tokens

This step checks for the presence of any contracted tokens in a given T–H pair and replaces the contracted tokens with their corresponding expanded forms. A few such tokens and their expanded forms are listed in Table 2.

Table 2: Contracted tokens and their expanded forms

Contracted tokens	Expanded forms
I’ve	I have
we’ll	we will
can’t	cannot
won’t	will not
we’d	we would

3.1.5 Merging of multiple sentences

Generally the length of the text is greater than the hypothesis and it is very often the case that the text consists of multiple sentences resulting in a short paragraph. This pre-processing step merges all those sentences in the paragraph by removing the intermediate end of sentence markers (i.e. full stops ‘.’) by hyphens (‘–’). An example text fragment from the RTE dataset is given below to illustrate the process.

Text: Today about 75 percent of people live in cities or towns. Art and culture are an important part of France.

After merging the two sentences by a hyphen, it takes the following form.

Text: Today about 75 percent of people live in cities or towns – Art and culture are an important part of France.

As mentioned earlier, the proposed RTE method is based on dependency graph matching. In order to explain the significance of this pre-processing step, the outputs of the Stanford dependency parser[1] before and after merging the sentences are presented in Table 3.

[1] http://nlp.stanford.edu/software/stanford-dependencies.shtml

Table 3: Output of Stanford dependency parser

Before merging	After merging
[tmod(live-7, Today-1) quantmod(75-3, about-2) num(percent-4, 75-3) nsubj(live-7, percent-4) prep_of(percent-4, people-6) root(ROOT-0, live-7) prep_in(live-7, cities-9) prep_in(live-7, towns-11) conj_or(cities-9, towns-11)] [nsubj(part-7, Art-1) conj_and(Art-1, culture-3) nsubj(part-7, culture-3) cop(part-7, are-4) det(part-7, an-5) amod(part-7, important-6) root(ROOT-0, part-7) prep_of(part-7, France-9)]	[dep(live-7, Today-1) quantmod(75-3, about-2) num(percent-4, 75-3) nsubj(live-7, percent-4) prep_of(percent-4, people-6) root(ROOT-0, live-7) prep_in(live-7, cities-9) nn(Art-12, towns-11) prep_in(live-7, Art-12) conj_or(cities-9, Art-12) nsubj(part-18, culture-14) cop(part-18, are-15) det(part-18, an-16) amod(part-18, important-17) conj_and(live-7, part-18) prep_of(part-18, France-20)]

The numbers next to the tokens in the dependency triplets play a vital role in generating the dependency graph structure (cf. Section 3.6). It can be noticed from the first column in Table 3 that for each sentence in the paragraph, the token numbers in the generated triplets start again from 0 at the ROOT. This duplication of the token numbers creates problems while combining these dependency triplets to form the dependency graph. To eliminate this problem of duplicate token numbers, we merge the sentences by removing the intermediate end of sentence markers by hyphens. After merging the sentences, the Stanford dependency parser produces the outputs as shown in the second column in Table 3. Since the token numbers are unique in this case, the dependency graph generation becomes easy.

Due to the ambiguous presence of dot (.) in acronyms, decimal fraction, abbreviated names, urls etc., it is hard to detect which dots signify the full stops (.) or the end markers of the sentences. Therefore, we cannot directly replace the occurrence of the dots by hyphens (–). In order to resolve this ambiguity, the sentences are first parsed by the dependency parser to identify where a particular sentence is ended. A few example T–H pairs with such ambiguous presence of dots are presented in Table 4 with the dotted tokens being highlighted in bold fonts.

Table 4: Example of sentences with ambiguous dots

Sentence	Tokens with ambiguous dots
Crude oil for April delivery traded at $**37.80** a barrel, down 28 cents.	37.80 (decimal fraction)
Larry Lawrence is the head of the **U.S.** Embassy in Switzerland.	U.S. (acronym)
John **J.** Famalaro is accused of having killed Denise **A.** Huber.	J. , A. (abbreviated names)
My Global Image LLC, announces the future of online networking through video streaming technology with the launch of **www.HelloDemoTour.com**	URL

3.2 Parts-Of-Speech tagging

The processed T–H pairs are then POS-tagged by a POS tagger. This phase plays a significant role in the subsequent stemming phase described in subsection 3.4.

3.3 Dependency Parsing

This phase takes each of the text fragments of the T–H pair, individually, as input and produces a set of dependency triplets as output. The dependency parser produces the triplets in the form of $R(W_1, W_2)$. A dependency triplet $R(W_G, W_D)$ represents a dependency relation (R) between two words - the governor (W_G) and the dependent (W_D).

3.4 Stemming

After the T–H pair is parsed, the generated tokens and their corresponding POS tags are fed into this phase. The stemming operation is performed in two steps.

In the first step, the Stanford stemmer[2] is invoked to produce the stem of each token. However, it was noticed that tokens belonging to several POS categories after being passed to the Stanford stemmer remain same as they were before being subjected to stemming. Therefore, for those categories of POS, the tokens are then passed on to another module RiTa[3] to obtain the true stems.

An example from the RTE3 devset is provided in Example 4 to show that the actual problem arises with the POS tagger rather than the stemmer itself. Finally, the RiTa stemmer is invoked for obtaining the actual stem.

Example 4:

<pair id="231" entailment="YES" task="IR" length="short" >
<t>Catastrophic **floods** in Europe endanger lives and cause human tragedy as well as heavy economic losses.</t>
<h>**Flooding** in Europe causes major economic losses.</h>
</pair>

The token ‘floods’ as highlighted in the text of the above pair belongs to the POS category NNS as tagged by the Stanford POS tagger[4] for which the Stanford stemmer produces the correct stem ‘flood’. The token ‘Flooding’ in the hypothesis is tagged as NN by the POS tagger. The Stanford stemmer does not produce the actual stem ‘flood’ for this token; rather the token ‘Flooding’ remains same after stemming operation. Therefore, the RiTa stemmer is invoked in the next step to get the true stem ‘flood’.

3.5 Lexical Alignment

This phase tries to lexically align each stemmed token W_H^i of the hypothesis with some token(s) W_T^j of the text. WordNet 2.1[5] was used as the lexical database and RiTa was employed to extract the lexical relations from the underlying database. Lexical alignment was carried out on the basis of a set of lexical relations listed in Table 5.

Table 5: WordNet Relations

Relation#	WordNet Relation
WR1	Stem match / direct match
WR2	Synonym
WR3	Hypernym
WR4	Hyponym/Troponym
WR5	Derivationally Related forms
WR6	Entailment

[2] http://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

[3] https://rednoise.org/rita/reference/RiWordNet.html

[4] http://nlp.stanford.edu/software/tagger.shtml

[5] https://wordnet.princeton.edu/

WR7	Causes
WR8	Antonyms

The output of this lexical alignment process is a set of token pairs (W_H^i, W_T^j) where the hypothesis token W_H^i is aligned with the text token W_T^j by any of the above mentioned WordNet relations.

An example is presented below to demonstrate the lexical alignment process.

Let us consider the following pair of sentence.

Example 5:

T: The students learn good morals from the teacher.

H: The person teaches good lessons to the pupils.

Figure. 2 shows the stemmed tokens of the two sentences and their corresponding alignment. The arrows represent the alignments and the different colours represent the different WordNet relations by which the text and hypothesis tokens are aligned. Table 6 shows the aligned T–H token pairs along with the matching criteria (i.e., WordNet relations). It is evident from Figure. 2 that a hypothesis token can be aligned with more than one token in the text, and vice-versa. The one-to-one mappings are indicated by thick arrows and the one-to-many and many-to-one mappings are indicated by dashed arrows in Figure 2.

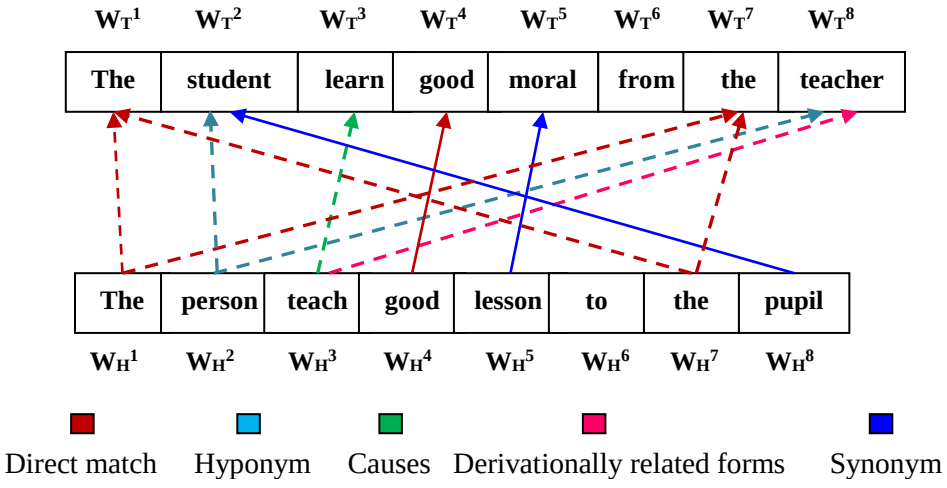

Figure. 2. Lexical alignment between text and hypothesis tokens

Table 6: Alignment of the T–H token pairs through WordNet relations

Hypothesis tokens (W#)	Aligned text tokens (W'#)	WordNet Relations
W_H^1	W_T^1	WR1
	W_T^7	WR1
W_H^2	W_T^2	WR4
	W_T^8	WR4
W_H^3	W_T^3	WR7
	W_T^8	WR5
W_H^4	W_T^4	WR1

W_H^5	W_T^5	WR2
W_H^6	Nil	–
W_H^7	W_T^1	WR1
	W_T^7	WR1
W_H^8	W_T^2	WR2

3.6 Dependency Graph representation

After the text and the hypothesis are dependency parsed, the generated triplets are combined together to form a dependency graph structure for both the text and the hypothesis. Each triplet in form of $R(W_1, W_2)$ is represented by a directed edge labelled by the relation (R) from the node containing the governor (W_1) to the node containing the dependent word (W_2) in the dependency graph.

Figure. 3.a depicts a particular branch of the dependency graph. Since a particular node may have multiple predecessors, we considered a dependency graph instead of a dependency tree. Figure. 3.b shows that a node (W) in the graph can have multiple predecessors as well as multiple successors. For the sake of simplicity, only 2 predecessors (W_1 and W_2) and 2 successors (W_3 and W_4) of the node W are depicted in the figure and the rest of the links are shown in dotted lines. In general, each node can have N (where $N \geq 0$) number of predecessors and/or successors resulting in a complex graph structure.

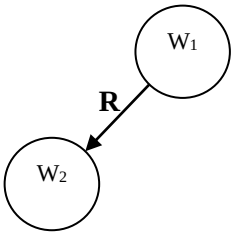

Figure. 3.a. Representation of the triplet $R(W_1, W_2)$ in the dependency graph

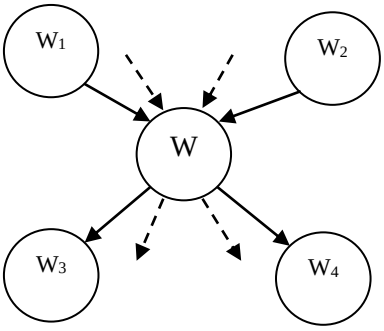

Figure. 3.b. Graph Structure

3.7 Comparing Dependency Graphs

Once the dependency graphs are constructed for the text (T) and the hypothesis (H), this module compares the two dependency graphs to determine whether H can be inferred from the corresponding graph structure of T. To accomplish this, each hypothesis triplet $R(W_1, W_2)$ is compared with every text triplets to search for a corresponding match. If a match is found according to any of the matching criterions stated below, a matching score in the range of [-1, 1] is assigned to the child node W_2 of that particular triplet. If none of the matching rules is satisfied, a score of 0 is assigned to W_2. This graph comparison module can be categorized into the following two types - lexical triplet matching module and semantic similarity module, which are described in the following subsections.

Before going into the detailed description of several categories of matching rules, the different score components of each hypothesis node of the dependency graph are presented below. Each hypothesis node has the following components.

- Vector of predecessor_score (*P*)
- Average of the predecessor_score vector (*A*)
- Child_score (*C*)
- Total_score (*T*)

This particular graph comparison module assigns a matching score to the *P* vector component of every hypothesis node. Apart from these 4 score components, each hypothesis node also contains 2 bits: an antonym bit (ant_bit) and a negation bit (neg_bit). Both of them are set to 0 initially. The significance of each of the 4 scoring components and the 2 specified bits are explained in subsection 3.8 where the dependency graph traversal process is described.

3.7.1 Lexical Triplet Matching Module

The lexical triplet matching module considers matching of the governor and the dependent words at the lexical level. This module can be broadly classified into the following two categories.

- **Single triplet dependency (STD):** A hypothesis dependency triplet is inferred from only one text dependency triplet.
- **Joint triplet dependency (JTD):** A hypothesis dependency triplet is inferred jointly from two or more text dependency triplets, or, one text dependency triplet infers two or more dependency triplets.

- **Single triplet dependency** For every dependency triplet of the hypothesis, the STD category tries to identify one matching triplet in the text. The matching of the hypothesis triplets with the text triplets is carried out in the present study following several rules. The matching rules which have been developed under this category are presented below. These rules, although have been mined by analysing the RTE datasets, are largely universal and can be applied to any sentence pair irrespective of the nature of the dataset.

Rule 1. This rule searches for a complete matching triplet. If the two nodes W_1 and W_2 of a hypothesis triplet $R^H(W_1,W_2)$ are lexically aligned with two nodes W_1' and W_2' respectively of a text triplet $R^T(W_1',W_2')$ and the relations R^T and R^H are identical, then the two triplets are identical to each other. Upon finding such a complete match, a matching score of 1 is assigned to the *P* vector component of the child node W_2 for that particular arc in the hypothesis dependency graph. Figure. 4 depicts this rule and a few examples from the RTE devsets are presented in Table 7 to illustrate the matching rule.

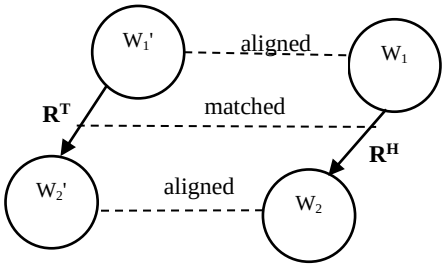

Figure. 4. Rule 1

Table 7: Examples satisfying matching rule 1

R^T and R^H	T–H pairs	Dependency triplets	Lexically aligned tokens	
			Token pair	Related By

nsubj	T: Ostriches **put** their heads into the sand to avoid the wind. H: Ostriches **bury** their heads in the sand.	nsubj(**put**-2, Ostriches-1) nsubj(**bury**-2, Ostriches-1)	put-bury	hypernym
nsubjpass	T: Yemen, too, was **reunified** in 1990. H: Yemen was **reunited** in 1990	nsubjpass(**reunified**-6,Yemen-1) nsubjpass(**reunited**-3, Yemen-1)	reunify-reunite	synset
dobj	T: Trained volunteers are **collecting** baseline data on water quality. H: Volunteers **gather** baseline data.	dobj(**collecting**-4, data-6) dobj(**gather**-2, data-4)	collect-gather	synset

Rule 2. If for any hypothesis triplet $R^H(W_1,W_2)$, the two tokens W_1 and W_2 lexically align with the text tokens W_1' and W_2' respectively connected by a text triplet $R^T(W_1',W_2')$ and the relations R^T and R^H are equivalent to each other belonging to this category of matching rule, then a matching score of 1 is assigned to the *P* vector component of the node W_2 in the hypothesis dependency graph. A total of 75 such equivalent relation pairs of this category were identified by a thorough analysis of RTE1, RTE2 and RTE3 devsets. A few examples satisfying this category are presented in Table 8. The highlighted tokens in bold fonts are lexically aligned with each other by some WordNet relations as listed in Table 5 other than WR1 and those relations are mentioned in the last column of Table 8. The matching rule is depicted in Figure. 5.

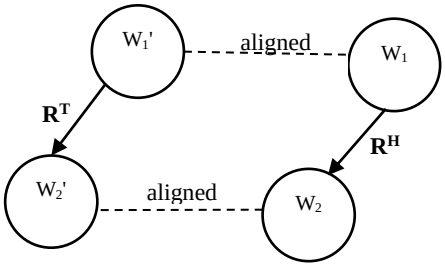

Figure. 5. Rule 2

Table 8: Examples satisfying matching rule 2

R^T	**R^H**	**T–H pairs**	**Equivalent dependency triples**	**Related By**
Nsubjpass	dobj	T: Lennon was **murdered** by Mark David Chapman outside the Dakota. H: Mark David Chapman **killed** Lennon.	nsubjpass(**murdered**-3, Lennon-1) dobj(**killed**-4, Lennon-5)	Hypernym
prep_of	nsubjpass	T: Hydroponics is the **growth** of plants in a substance other than soil with water. H: Plants are **grown** in substances other than soil.	prep_of(**growth**-4, plants-6) nsubjpass(**grown**-3, Plants-1)	Derivationally Related
Nsubj	nn	T: The recent 14% hike in third class postage rates, accompanied by simultaneous double-digit paper **price increases**, has hit smaller catalogers especially hard. H: The **cost** of paper is **rising**.	nn(**increases**-18, price-17) nsubj(**rising**-6, cost-2)	Synonym
prep_in	amod	T: The biggest newspaper in **Norway**, Verdens Gang, prints a letter to the editor written by Joe Harrington and myself. H: Verdens Gang is a **Norwegian** newspaper.	prep_in(newspaper-3, **Norway**-5) amod(newspaper-6, **Norwegian**-5)	Derivationally Related
Dobj	prep_of	T: Bukowski attracted the attention of John Martin who **founded** Black Sparrow Press	dobj(**founded**-9, Press-12)	Derivationally Related

		in 1965 specifically to publish him. H: John Martin is the **founder** of Black Sparrow Press.	prep_of(**founder**-5, Press-9)	

Rule 3. If the two tokens W_1 and W_2 connected by a hypothesis triplet $R^H(W_1,W_2)$ are aligned to the text tokens W_1' and W_2' respectively connected by a text triplet $R^T(W_2',W_1')$ and the relations R^T and R^H are equivalent to each other under this particular category of matching rule, then a matching score of 1 is assigned to the *P* vector component of the child node W_2 of the hypothesis relation. 26 such equivalent relation pairs were generated by analysing the RTE development sets. A few examples adhering to this rule are presented in Table 9. Figure. 6 depicts the matching rule.

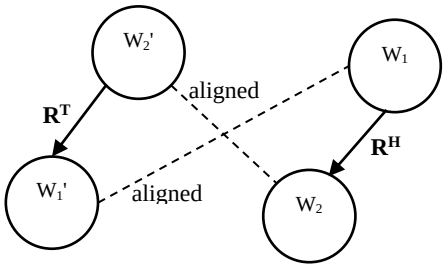

Figure. 6. Rule 3

Table 9: Examples satisfying matching rule 3

R^T	R^H	T–H pairs	Equivalent dependency triples
partmod	nsubjpass	T: This case of rabies in western Newfoundland is the first case confirmed on the island since 1989. H: A case of rabies was confirmed.	partmod(case-11,confirmed-12) nsubjpass(confirmed-6, case-2)
amod	dobj	T: Democrat Culbert L. Olson, elected governor of California in 1938, was a loyal supporter of Roosevelt's New Deal. H: Democrat Culbert L. Olson was elected governor of California.	amod(governor-7, elected-6) dobj(elected-6, governor-7)
amod	nsubjpass	T: Environment Minister David Anderson announced his decision regarding the environmental assessment of the proposed Toulnustouc Hydroelectric Project. H: A hydroelectric project is proposed or is under construction.	amod(Project-17, proposed-14) nsubjpass(proposed-5, project-3)
rcmod	nsubjpass	T: In clashes between Israeli forces and gunmen, one Palestinian was killed and 10 wounded. H: A Palestinian was killed and other people were wounded.	nsubjpass(killed-4, Palestinian-2) rcmod(Palestinian-10, killed-12)
partmod	dobj	T: Eye injuries caused by fireworks are extremely serious and can permanently damage eyesight. H: Fireworks may cause serious injuries.	partmod(injuries-2, caused-3) dobj(cause-3, injuries-5)

Rule 4. This rule is specifically designed for handling antonyms. There are two subcategories under this rule, which are described below.

- **Category 1.** If for a hypothesis triplet $R^H(W_1,W_2)$, there exists a text triplet $R^T(W_1^A,W_2')$ such that the word pair W_1–W_1^A are antonyms of each other, the word pair W_2–W_2' are lexically aligned and the relations R^H–R^T are equivalent to each other according to matching rule 2 as stated earlier, then a score of -1 is assigned

to the *P* vector component of the child node W_2 of the hypothesis triplet. Similarly, if the hypothesis triplet $R^H(W_1,W_2)$ matches with a text triplet $R^T(W_2', W_1^A)$, where the relations R^H and R^T are equivalent to each other as per rule 3, in that case also a score of -1 is assigned to the *P* vector component of the node W_2 of the hypothesis dependency graph. As this particular category indicates contradiction, a negative score is assigned. Figure. 7.a and Figure. 7.b depict the above mentioned two cases pictorially. A few examples under this category are presented in Table 10.

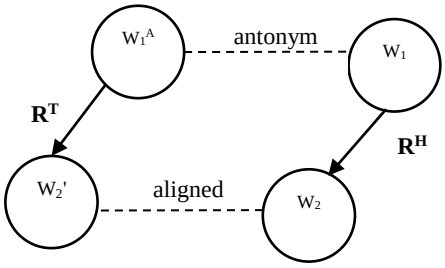

Figure. 7.a.Case 1 of category 1 of rule 4

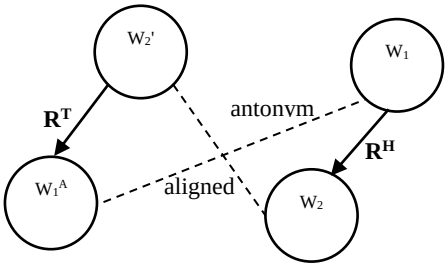

Figure. 7.b. Case 2 of category 1 of rule 4

Table 10: Examples satisfying category 1 of matching rule 4

R^T	R^H	T–H pairs	Dependency triples
nsubj	nsubj	T: Doug Lawrence **bought** the impressionist oil landscape by J. Ottis Adams in the mid-1970s at a Fort Wayne antiques dealer. H: Doug Lawrence **sold** the impressionist oil landscape by J. Ottis Adams.	nsubj(**bought**-3,Lawrence-2) nsubj(**sold**-3, Lawrence-2)
nsubj	amod	T: Newspapers choke on **rising** paper costs and falling revenue. H: The cost of paper is **falling**.	amod(costs-6, **rising**-4) nsubj(**falling**-6, cost-2)

- **Category 2.** The hypothesis triplet $R^H(W_1,W_2)$ is searched for a corresponding matching text triplet $R^T(W_1',W_2^A)$ such that tokens W_2–W_2^A are antonyms, W_1–W_1' are lexically aligned and the relation pair R^T–R^H is equivalent according to rule 2. If such a hypothesis–text triplet pair can be found, then the ant_bit of W_2 is set to 1. The same action is performed where the hypothesis triplet $R^H(W_1,W_2)$ matches with a text triplet $R^T(W_2^A, W_1')$, where the relation pair R^T–R^H is equivalent as per matching rule 3. Figure. 8.a and 8.b depict these two cases. Examples satisfying this category are presented in Table 11. It is to be noted that no examples were found in the RTE development sets which match with the case depicted in Figure. 8.b.

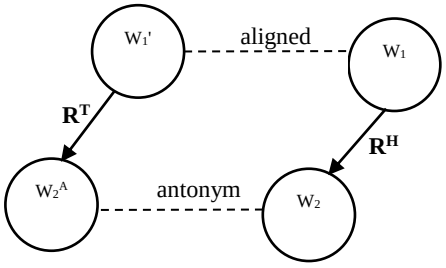

Figure. 8.a. Depicting case 1 of category 2 of rule 4

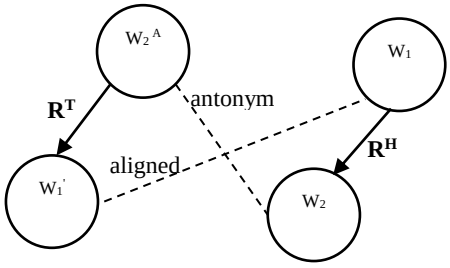

Figure. 8.b. Depicting case 2 of category 2 of rule 4

Table 11: Examples satisfying category 2 of matching rule 4

R^T	R^H	T–H pairs	Dependency triples
amod	amod	T: He bought a **new** car. H: He bought an **old** car.	amod(car-5, **new**-4) amod(car-5, **old**-4)

nsubj	nsubj	T: The **boy** won the game. H: The **girl** won the game.	nsubj(won-3, **boy**-2) nsubj(won-3, **girl**-2)

Rule 5. This particular rule was developed for handling negations.

- **Category 1.** For each dependency triplet $R^H(W_1,W_2)$, all the text triplets are searched for a matching pair $R^T(W_1',W_2')$ such that the token pairs W_1–W_1' and W_2–W_2' are lexically aligned with some relations other than the antonym relation and the relation pair R^H–R^T is equivalent according to matching rule 2. Now, if any one of the two triplets $R^H(W_1,W_2)$ and $R^T_1(W_1',W_2')$ has a negation relation R_N associated with it, then the neg_bit value of the parent node W_1 of the particular branch of the hypothesis dependency tree is set to 1. Figure. 9.a. and 9.b illustrate this category 1 of rule 5. W^N denotes the node containing the negative terms such as "not", "never", "no", etc.
  However, if both of the text and hypothesis triplets are associated with a negation relation, then the negation rule is not applicable and a matching score of 1 is assigned to the *P* vector component of the child node W_2 of that particular branch of the hypothesis dependency tree. Figure. 9.c depicts this particular case.

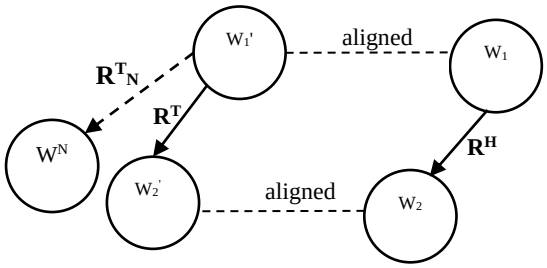

Figure. 9.a. Case 1 of category 1 of rule 5

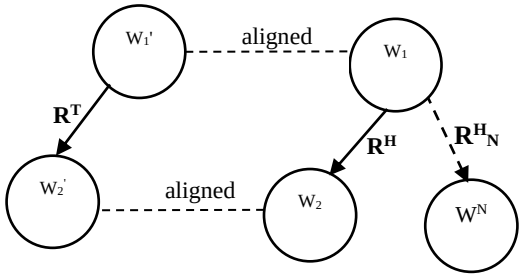

Figure. 9.b. Case 2 of category 1 of rule 5

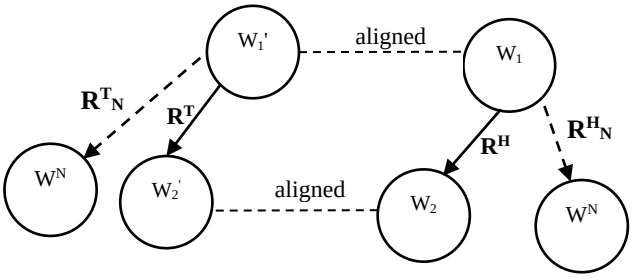

Figure. 9.c. Rule 5 is not applied

- **Category 2.** If a hypothesis dependency triplet $R^H(W_1,W_2)$ matches with a text triplet $R^T(W_2',W_1')$ such that the token pairs W_1–W_1' and W_2–W_2' are lexically aligned through some relations other than the antonym relation and the relation pair R^H–R^T is equivalent according to matching rule 3 and any of the 2 triplets is involved in a negation relation R_N, then the neg_bit value of the parent node W_1 of that particular branch of the hypothesis dependency tree is set to 1. Figure. 10.a and 10.b. illustrate the category 2 of this rule diagrammatically. A few examples satisfying this rule are presented in Table 12.

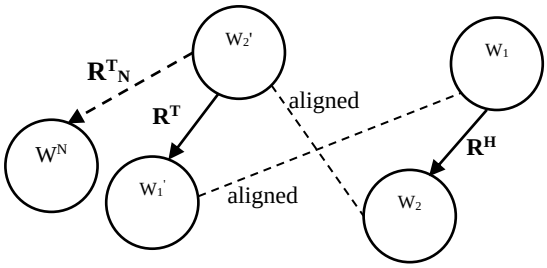

Figure. 10.a. Case 1 of category 2 of rule 5

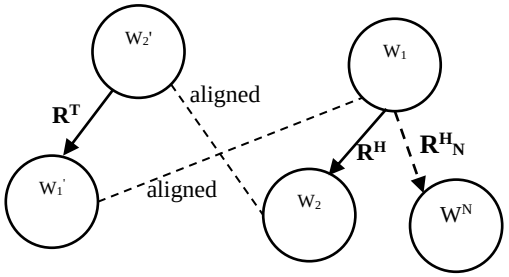

Figure. 10.b. Case 2 of category 2 of rule 5

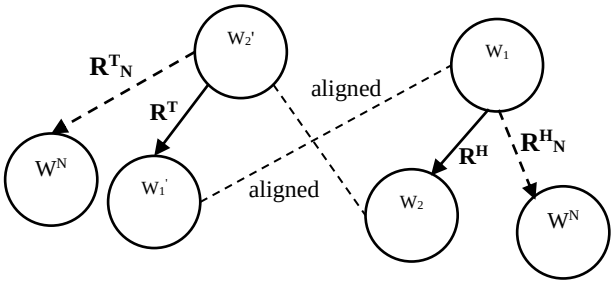

Figure. 10.c. Rule 5 is not applied

Table 12: Examples satisfying matching rule 5

R^T	R^H	T–H pairs	Dependency triples
amod	amod	T: Clinton's new book is **not** big seller here. H: Clinton's book is a big seller.	neg(seller-8,**not**-6) amod(seller-8, big-7) amod(seller-7, big-6)
dobj	dobj	T: Mohandas Karamchand Gandhi **never** received the Nobel Peace Prize, though he was nominated for it five times between 1937 and 1948. H: Mohandas received the Nobel Prize in 1989.	neg(received-5, **never**-4) dobj(received-5, Prize-9) dobj(received-2, Prize-5)
nsubjpass	dobj	T: Lennon was **murdered** by Mark David Chapman outside the Dakota. H: Mark David Chapman did **not kill** Lennon.	nsubjpass(**murdered**-3, Lennon-1) dobj(**kill**-6, Lennon-7) neg(**kill**-6, **not**-5)
amod	nsubj	T: Newspapers choke on falling paper costs. H: The cost of paper is **not** falling.	amod(costs-6, falling-4) nsubj(falling-7, cost-2) neg(falling-7, **not**-6)
det	–	T: **No** weapons of Mass Destruction Found in Iraq Yet. H: Weapons of Mass Destruction Found in Iraq.	det(weapons-2, **No**-1) –

Figure. 9.c. and 10.c depict the cases where this rule is not applicable. Let us consider the example T–H pair below.

Example 6:

T: He did **not buy** a car.

H: He did **not purchase** a car.

Since the word pair *buy–purchase* in Example 6 is lexically aligned by the synonym relation in the WordNet and both of them are negated, both the text and the hypothesis express the same meaning. Therefore rule 5 is not

applicable to such cases where both the text and hypothesis dependency triplets are associated with negation relations.

One more example sentence pair with negation relations is presented in Example 7.

Example 7:

T: Everything is permanent in life.

H: Nothing is permanent in life.

For the given pair of Example 6, the dependency parser generates two triplets nsubj(permanent-3, Everything-1) and nsubj(permanent-3, Nothing-1) for the text and hypothesis respectively. The token "nothing" indicates a negation relation associated to the token "permanent" in the hypothesis. However, the text is free from any negation relation. Therefore according to the scoring mechanism of this rule, the neg_bit value of the hypothesis node containing the token "permanent" is set to 1.

In this way, the method can correctly handle the sentence pairs involving words like something–nothing, somebody–nobody, everyone–none, somewhere–nowhere, etc. These types of cases, although not found in the RTE datasets, are also handled to improve completeness and robustness of the method.

Rule 6. This rule was synthesized for the cases where negations and antonym relations are combined.

- **Category 1 and Category 2.** If for any hypothesis dependency triplet $R^H(W_1,W_2)$, a triplet $R^T(W_A',W_2')$ or $R^T(W_1',W_A')$ is found in the text where the token pair W_1–W_A' or W_2–W_A' are antonyms of each other, the W_2–W_2' or W_1–W_1' word pairs are lexically aligned with each other, the relation pair R^H–R^T is equivalent according to matching rule 2 and a negation relation R_N is attached to any of the text or hypothesis triplet, then they convey the same meaning and a matching score of 1.0 is assigned to the *P* vector component of the hypothesis node W_2 that corresponds to the triplet $R^H(W_1, W_2)$. Figure. 11.a, 11.b. and Figure. 12.a, 12.b depict these categories diagrammatically. It is to be noted that no T–H pair was found in the RTE datasets which satisfies this category of matching rule. The rule has been designed for the completeness of the method and to make it general to the highest possible extent. Therefore, some random T–H pairs have been provided in Tables 13 and 14 satisfying categories 1 and 2.

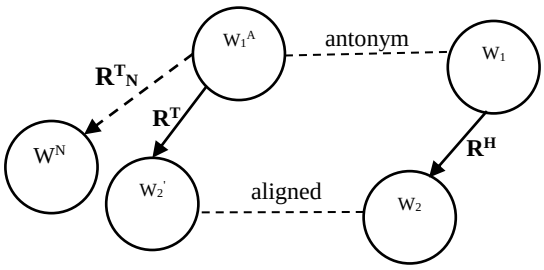

Figure. 11.a. Depicting case 1 of category 1 of rule 6

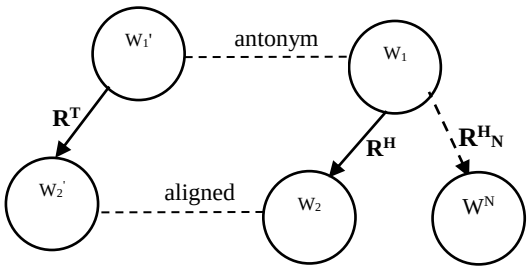

Figure. 11.b. Depicting case 2 of category 1 of rule 6

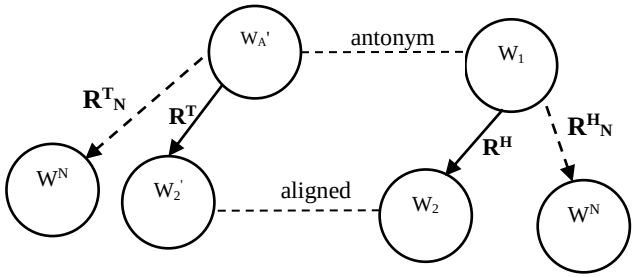

Figure. 11.c. Depicting case 3 of category 1 of rule 6

Table 13: Examples satisfying category 1 of rule 6

Pair Id#	R^T	R^H	T–H pairs	Dependency triples
1	nsubj	nsubj	T: We are **not interested** about the matter. H: We are **uninterested** about the matter.	nsubj(**interested**-4, We-1) neg(**interested**-4, **not**-3) nsubj(**uninterested**-3, We-1)
2	nsubj	nsubj	T: They are **unable** to do the task. H: They are **not able** to do the task.	nsubj(**unable**-3, They-1) nsubj(**able**-4, They-1) neg(**able**-4, **not**-3)
3	nsubj	nsubj	T: He did **not pass** in the exam. H: He **failed** in the exam.	nsubj(**pass**-4, He-1) neg(**pass**-4, **not**-3) nsubj(**failed**-2, He-1)
4	amod	amod	T: He is an **honest** person. H: He is **not a dishonest** person.	amod(**honest**-4, person-5) amod(**dishonest**-5, person-6) neg(**dishonest**-5, **not**-3)
5	nsubj	nsubj	T: John **loves** Mary. H: John does **not hate** Mary.	nsubj(**loves**-2, John-1) nsubj(**hate**-4, John-1) neg(**hate**-4, **not**-3)

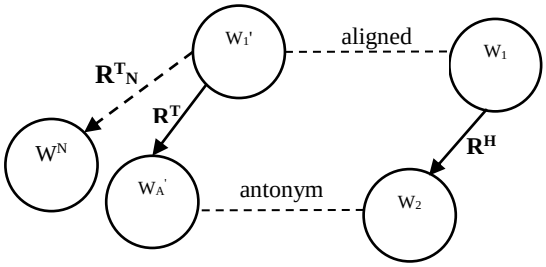

Figure. 12.a. Depicting case 1 of category 2 of rule 6

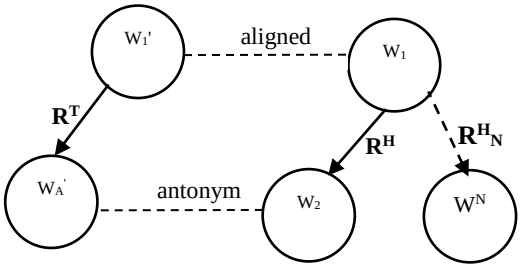

Figure. 12.b. Depicting case 2 of category 1 of rule 6

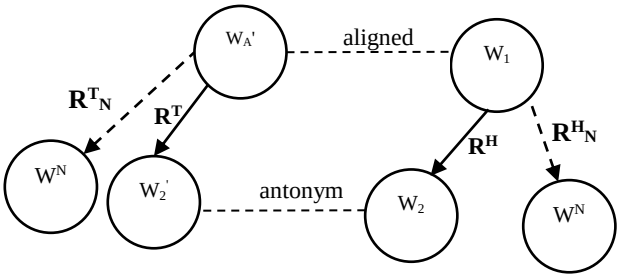

Figure. 12.c. Depicting case 3 of category 2 of rule 6

Table 14: Examples satisfying category 2 of rule 6

Pair Id#	R^T	R^H	T–H pairs	Dependency triples
1	advmod	advmod	T: I help them **unselfishly.** H: I do **not** help them **selfishly.**	advmod(help-2, **unselfishly**-4) advmod(help-4, **selfishly**-6) neg(help-4, **not**-3)
2	amod	amod	T: This would **not** be an **ethical** task to do. H: This would be an **unethical** task.	amod(task-7, **ethical**-6) neg(task-7, **not**-3) amod(task-6, **unethical**-5)

However, if both the text and hypothesis triplets are associated with negation relations, then it indicates contradiction and according to the rule of handling antonyms a matching score of -1.0 is assigned to the *P* vector component of the hypothesis node W_2 (illustrated in Figure. 11.c and 12.c). Let us consider the following T–H pair.

Example 8:

T: He did **not pass** in the exam.

H: He did **not fail** in the exam.

In this pair of text and hypothesis, the words **pass** and **fail** are antonyms of each other and both of them are negated. Therefore unlike the pairs presented in Tables 13 and 14, this T–H pair expresses the opposite meaning. Since the token **pass** on applying negation becomes **fail** and the token **fail** on getting negated becomes **pass**, the T–H pair in Example 8 is equivalent to the following pair:

T: He **failed** in the exam.

H: He **passed** in the exam.

Therefore, according to rule 4 of handling antonym, a score of -1 is assigned to the *P* vector component of the word **pass** in the hypothesis dependency graph.

- **Category 3 and 4**. If for any hypothesis dependency triplet $R^H(W_1,W_2)$, a text triplet $R^T(W_2',W_A')$ or $R^T(W_A',W_1')$ can be found such that the tokens W_1–W_A' or W_2–W_A' are aligned by antonym relation, W_2–W_2' or W_1–W_1' are aligned with each other by any other lexical relation, the relations R^H and R^T are equivalent to each other as per matching rule 3 and any of the text or hypothesis triplets is associated with a negation relation R_N, then a matching score of 1.0 is assigned to the *P* vector component of the hypothesis node W_1 corresponding to the triplet $R^H(W_1,W_2)$. The Figure. 13.a, 13.b and Figure. 14.a, 14.b illustrate the different cases of the categories pictorially. It is to be noted that no examples were found for category 4 depicted in Figure. 14.a and 14.b. Table 15 shows some T–H pairs satisfying this category.

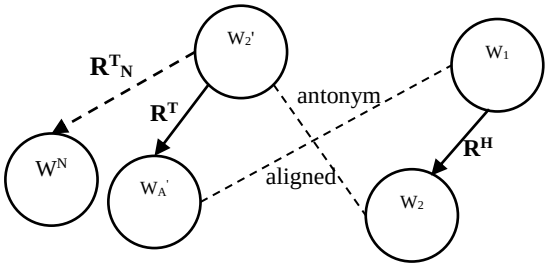

Figure. 13.a. Depicting case 1 of category 3 of rule 6

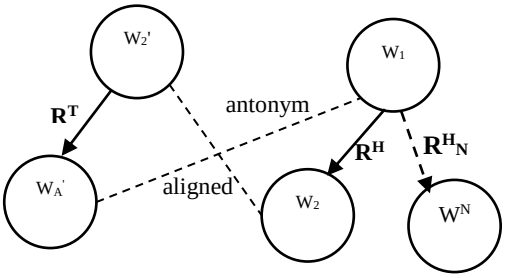

Figure. 13.b. Depicting case 2 of category 3 of rule 6

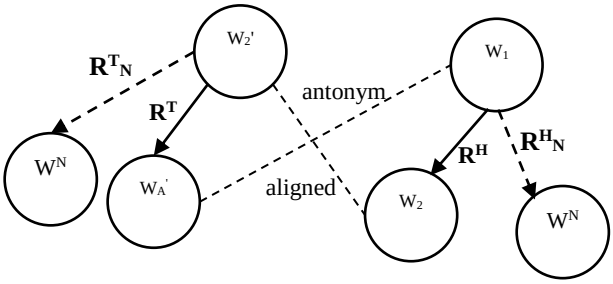

Figure. 13.c. Depicting case 3 of category 3 of rule 6

Table 15: Examples satisfying category 3 of rule 6

R^T	R^H	T–H pairs	Dependency triples
amod	nsubj	T: Newspapers choke on **falling** paper costs. H: The cost of paper is not **rising**.	amod(costs-6, **falling**-4) nsubj(**rising**-7, cost-2) neg(**rising**-7, not-6)
infmod	nsubjpass	T: The report catalogues 10 missed opportunities within the CIA and FBI to **uncover** pieces of the September 11 plot. H: Ten missed opportunities within the CIA and FBI are **not covered** in the report.	infmod(opportunities-6, **uncover**-13) nsubjpass(**covered**-11, opportunities-3) neg(**covered**-11, **not**-10)

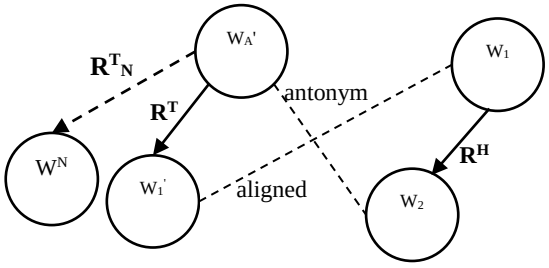

Figure. 14.a. Depicting case 1 of category 4 of rule 6

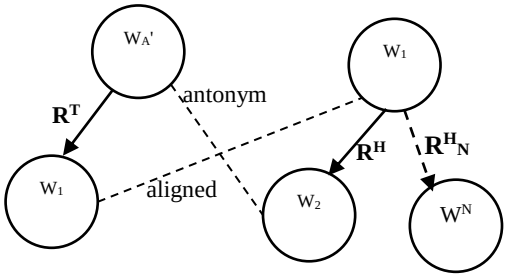

Figure. 14.b. Depicting case 2 of category 4 of rule 6

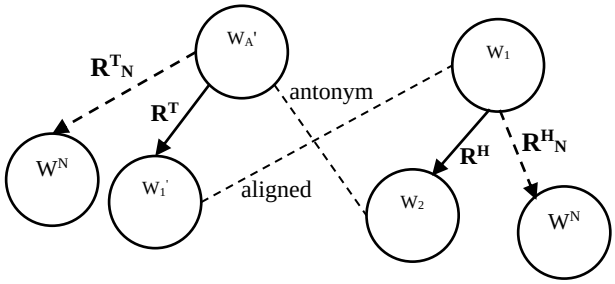

Figure. 14.c. Depicting case 3 of category 4 of rule 6

If both the text and hypothesis triplets are associated with a negation relation each, then a score of -1.0 is assigned to *P* vector component of the hypothesis node W_2 as it represents contradiction. Figure. 13.c and 14.c depict the cases. Table 16 presents some cases where this rule is not applicable.

Table 16: Examples where Rule 6 is not applicable

Pair id#	R^T	R^H	T–H pairs	Dependency triples

1	nsubj	nsubj	T: He did **not buy** a car. H: He **sold** a car.	nsubj(**buy**-4, He-1) neg(**buy**-4, **not**-3) nsubj(**sold**-2, He-1)
2	nsubjpass	nsubj	T: He was **not born** in California. H: He **died** in California.	nsubjpass(**born**-4, He-1) neg(**born**-4, **not**-3) nsubj(**died**-2, He-1)
3	dobj	dobj	T: She teaches her **brother**. H: She does **not** teach her **sister**.	dobj(teaches-2, **brother**-4) dobj(teach-4, **sister**-6) neg(teach-4, **not**-3)
4	nsubj	nsubj	T: John does **not love** Mary. H: John **hates** Mary.	nsubj(**love**-4, John-1) neg(**love**-4, **not**-3) nsubj(**hates**-2, John-1)

For the 1st pair presented in the above table, although *buy* and *sell* are antonyms of each other, however this rule is still not applicable. If *X did not buy a car*, then it cannot be said that *X sold a car*. Similarly for the 2nd case, if *X was not born in California*, then it does not imply that *X died in California*. For the third sentence pair, despite *brother* and *sister* being antonyms of each other, if *X teaches her brother*, then it does not entail that *X does not teach her sister*. They totally contradict each other in meaning.

In the last sentence pair (pair ID 4), although *love* and *hate* are antonyms of each other, still it cannot be implied that *A hates B* if *A does not love B*. If *A does not love B*, it does not always mean that *A hates B*, because *A may be neutral to B* or even *A may be unknown to B*. But the reverse of this particular case, pair ID 5 in Table 13 satisfies this rule. In the reverse direction, if *A loves B*, then obviously it implies *A does not hate B*.

Therefore, it is evident from the examples that verb pairs *buy-sell, die-born, love-hate* are not absolute antonyms as far as textual entailment is concerned. This rule can only be successfully applied to those T–H pairs where absolute antonym word pairs such as *pass-fail, able-unable, ethical-unethical, rise-fall*, etc. are negated as presented in Tables 13, 14 and 15. However, the lexical database WordNet 2.1 embedded in our method does not contain any relevant information by which we can distinguish the absolute antonyms from the others. Therefore, the method shows a limitation to correctly apply this matching rule to triplets containing absolute antonym pairs only.

Rule 7. A few insignificant relations were also identified by analyzing the RTE development sets. These are {det, expl, aux, auxpass, cop, mark, prt, predet}. If for any triplet $R^H(W_1,W_2)$ of the hypothesis, the relation R^H belongs to this relation set, then a matching score of 1 is assigned to the *P* vector component of the node W_2 of that arc in the dependency graph. These relations, although of less importance, if not considered and properly not taken care of, may result in inappropriate entailment scores leading to incorrect entailment decisions. A few example sentences involving dependency relations belonging to this category are presented in Table 17.

Table 17: Examples satisfying matching rule 7

R_H	Hypothesis	Dependency triples
det	A case of rabies was confirmed.	det(case-2, A-1)
expl	There is a territorial waters dispute.	expl(is-2, There-1)
auxpass	A civilian policeman was killed.	auxpass(killed-5, was-4)

- **Joint Triplet Dependency (JTD)** If for any triplet $R^H(W_1,W_2)$ of the hypothesis, no corresponding matching triplet is found in the text, this category tries to find two dependency triplets of the text that can jointly infer $R^H(W_1,W_2)$. Although more than two text triplets together can imply a single hypothesis triplet, we considered up to a maximum of 2 text triplets to keep the method less complex; however, the method is

generic and can be extended to consider more than two text triplets. On finding two text triplets jointly inferring the hypothesis dependency triplet $R^H(W_1,W_2)$, a matching score of 1 is assigned to the *P* vector component of the node W_1 of the hypothesis dependency graph. The JTD category can again be classified into 6 types each of which are discussed below with suitable diagrams and examples. After analysing the RTE development sets, we identified a number of relations which fall under a specific category which we refer to as Joint Dependency Compatible (JDC) relations. The significance of JDC relations is explained later.

- **Category 1**. For a particular triplet $R^H(W_1,W_2)$ of the hypothesis, this category tries to find two text triplets $R^T_1(W_1',W)$ and $R^T_2(W,W_2')$ so that the two tokens W_1 and W_2 in the hypothesis triplet are lexically aligned with the tokens W_1' and W_2', respectively, associated with the two text triplets. R^H is either identical or equivalent to R^T_1 according to Rule 2 of the STD category and R^T_2 must belong to the JDC category. Figure. 15 depicts this rule diagrammatically. A few examples satisfying this rule are presented in Table 18. The highlighted tokens in the table indicate that they are aligned to each other by some WordNet relations other than direct/stem match. For this particular case, '*kill*' and '*murder*' are synonyms to each other. The table shows which relations belong to JDC category and which relations are equivalent to R^H.

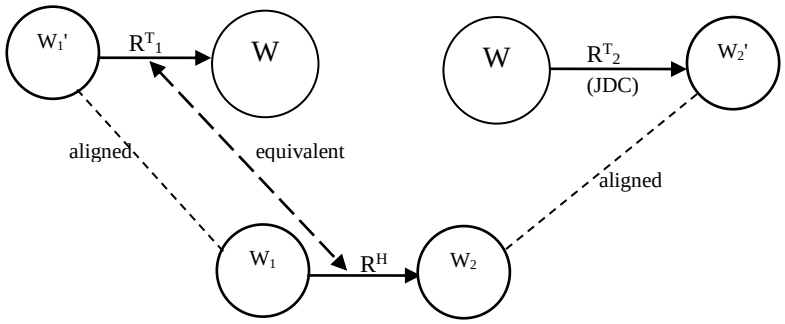

Figure. 15. Depicting Category 1 of JTD

Table 18: Examples satisfying category 1 of JTD

R^H	R^T_1	R^T_2 (JDC)	T–H pairs	Dependency triples
prep_in	prep_at	poss	T: The Rolling Stones kicked off their latest tour on Sunday with a concert at Boston's Fenway Park. H: The Rolling Stones have begun their latest tour with a concert in Boston.	prep_at(concert-13, Park-18) poss(Park-18, Boston-15) prep_in(concert-11, Boston-13)
nsubjpass	nsubj	appos	T: Alleged terrorists today, **killed** Dolores Hinostroza, the mayor of Mulqui district, shooting her five times. H: The mayor of Mulqui district was **murdered** with a firearm.	nsubj(**killed**-5, Hinostroza-7) appos(Hinostroza-7, mayor-10) nsubjpass(**murdered**-7,mayor-2)

- **Category 2.** This category searches all the text triplets to identify 2 text triplets $R^T_1(W_1',W)$ and $R^T_2(W, W_2')$ that can jointly infer a triplet $R^H(W_1,W_2)$ of the hypothesis. Here R^T_2 must be same or equivalent to R^H according to the matching criterion of Rule 2 of STD. Whereas R^T_1 should belong to the JDC category of relations. Diagrammatic representation of this category 2 type JTD is shown in Figure. 16 and an example T–H pair satisfying this category is presented in Table 19.

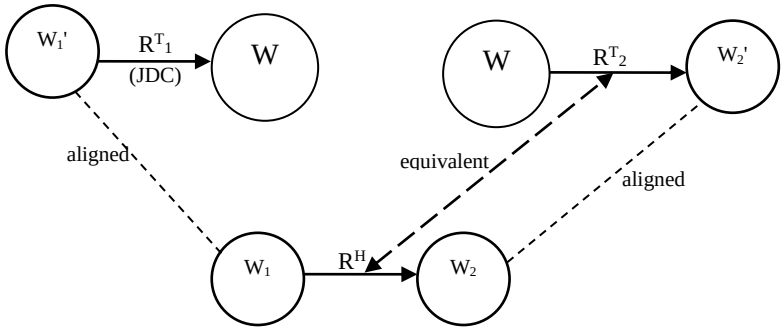

Figure. 16. Depicting Category 2 of JTD

Table 19: Example satisfying category 2 of JTD

R^H	R^{T_1}	R^{T_2} (JDC)	T–H pairs	Dependency triples
nsubj	prep_by	Appos	T: Norway's most famous painting, 'The Scream' by Edvard Munch, was recovered Saturday, almost three months after it was stolen from an Oslo museum. H: Edvard Munch painted 'The Scream'.	prep_by(Scream-9, Munch-13 appos(painting-5, Scream-9) nsubj(painted-3, Munch-2)

- **Category 3.** For a particular hypothesis dependency triplet $R^H(W_1,W_2)$, this category tries to find two text triplets $R^{T_1}(W_2',W)$ and $R^{T_2}(W,W_1')$ where R^H is equivalent to R^{T_2} according to rule 3 of STD and R^{T_1} belongs to JDC type relations. Figure. 17 depicts this category and an example T–H pair is provided in Table 20 to illustrate the matching rule.

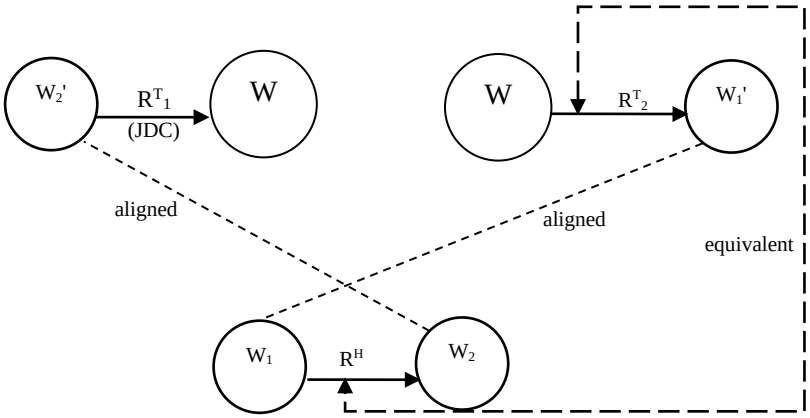

Figure. 17. Depicting Category 3 of JTD

Table 20: Example satisfying category 3 of JTD

R^H	R^{T_2}	R^{T_1}(JDC)	T–H pairs	Dependency triples
nsubjpass	amod	Appos	T: Democrat Culbert L. Olson, elected governor of California in 1938, was a loyal supporter of Roosevelt's New Deal. H: Democrat Culbert L. Olson was elected governor of California.	amod(governor-7, elected-6) appos(Olson-4, governor-7) nsubjpass(elected-6, Olson-4)

- **Category 4**. If two text triplets $R^{T_1}(W_2',W)$ and $R^{T_2}(W,W_1')$ can jointly infer a single hypothesis triplet $R^H(W_1,W_2)$, then category 4 is satisfied. Here R^H should be identical or equivalent to R^{T_1} as per Rule 3 of STD and R^{T_2} must belong to JDC category. This category is depicted in Figure. 18 and an example T–H pair satisfying this rule is presented in Table 21.

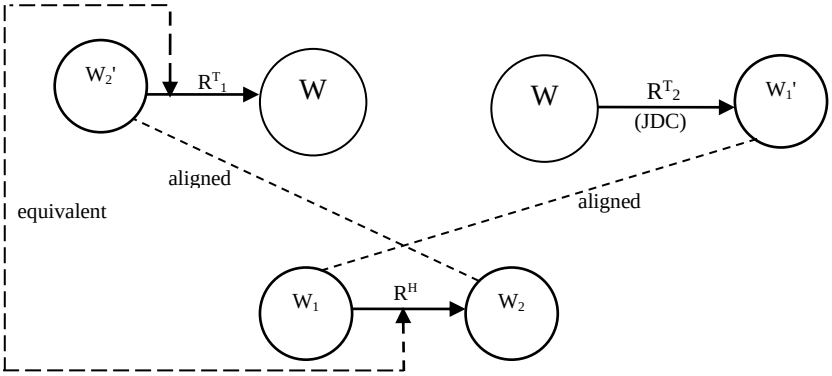

Figure. 18. Depicting Category 4 of JTD

Table 21: Example satisfying category 4 of JTD

R^H	R^T_1	R^T_2(JDC)	T–H pairs	Dependency triples
nsubj	appos	Appos	T: Black Tigers are selected for their discipline and their loyalty to Vilupillai Prabhakaran, 41, the elusive leader of the LTTE. H: Vilupillai Prabhakaran is the elusive leader of the LTTE.	appos(Prabhakaran-13, 41-15) appos(41-15, leader-19) nsubj(leader-6, Prabhakaran-2)

- **Category 5.** This category of JTD searches all the text triplets to find out two triplets $R^T_1(W,W_1)$ and $R^T_2(W,W_2')$ that can jointly infer a single hypothesis dependency triplet $R^H(W_1,W_2)$. Here R^H should be equivalent to any of R^T_1 or R^T_2 and the other one should belong to the JDC type relation. Figure. 19 diagrammatically depicts this rule and Table 22 presents some examples of T–H pairs satisfying this category.

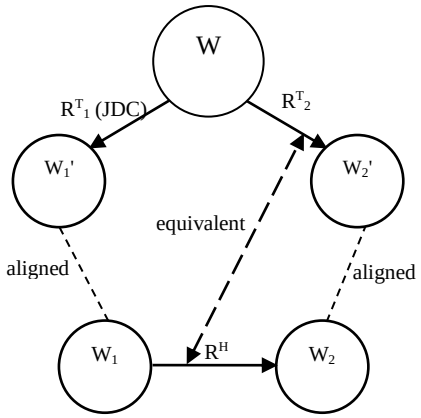

Figure. 19. Depicting Category 5 of JTD

Table 22: Examples satisfying category 5 of JTD

R^H	R^T_1	R^T_2(JDC)	T–H pairs	Dependency triples
prep_of	nn	Nn	T: In support of the Earth Island Institute rebuttal, Greenpeace founder Don White made the following - unfortunately prophetic - public statement on 8/18/93. H: Don White is the founder of Greenpeace.	nn(White-13, Greenpeace-10), nn(White-13, founder-11) prep_of(founder-5, Greenpeace-7)
Nn	nn	Nn	T: Herceptin was already approved to treat the sickest breast cancer patients, and the company said, Monday, it will discuss with federal regulators the possibility of prescribing the drug for more breast cancer patients.	nn(patients-11,breast-9), nn(patients-11, cancer-10) nn(cancer-8, breast-7)

			H: Herceptin can be used to treat breast cancer.	

- **Category 6.** This final category of JTD identifies 2 text triplets $R^{T}_1(W, W_1)$ and $R^{T}_2(W, W_2')$ that together imply a single triplet $R^{H}(W_1,W_2)$ of the hypothesis. The hypothesis tokens W_1 and W_2 are lexically aligned with the text tokens W_1' and W_2', respectively. Figure. 20 depicts this category. In this particular case, no equivalent relations to R^H or any Joint Dependency Compatible (JDC) relations are indicated. A few T–H pairs satisfying this category are presented in Table 23.

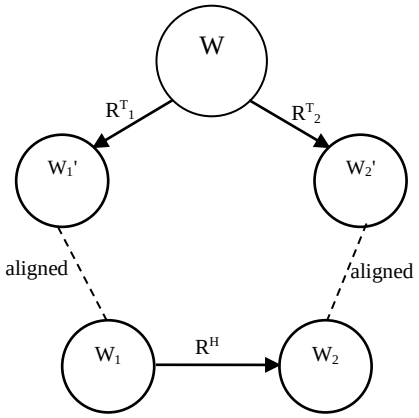

Figure. 20. Depicting Category 6 of JTD

Table 23: Examples satisfying category 6 of JTD

R^H	R^T_1	R^T_2	T–H pairs	Dependency triples
nsubj	dobj	nsubj	T: Blair has sympathy for anyone who has lost their lives in Iraq. H: Blair is sympathetic to anyone who has lost their lives in Iraq.	nsubj(has-2, Blair-1) dobj(has-2, sympathy-3) nsubj(sympathetic-3, Blair-1)
poss	dobj	nsubj	T: The Sears Tower has 110 stories compared to the twin buildings of 88 stories each. H: The twin buildings are 88 stories each, compared with the Sears Tower's 110 stories.	nsubj(has-4, Tower-3) dobj(has-4, stories-6) poss(stories-16, Tower-13)

- **Reverse JTD:** It may also be the case that two hypothesis dependency triplets $R^{H}_1(W_1, W)$ and $R^{H}_2(W, W_2)$ can be inferred from one text triplet $R^{T}(W_1',W_2')$. For each of the six categories of JTD presented earlier, we also consider the reverse case. However, due to space constraints, only the reverse case of category 1 JTD is depicted with diagram (Figure. 21) and Table 24 presents some example T–H pairs from the RTE datasets satisfying this reverse category. On satisfying this reverse category of JTD, a matching score of 1 is assigned to the *P* vector component of both the dependent nodes W and W_2 associated with the hypothesis triplets as stated above.

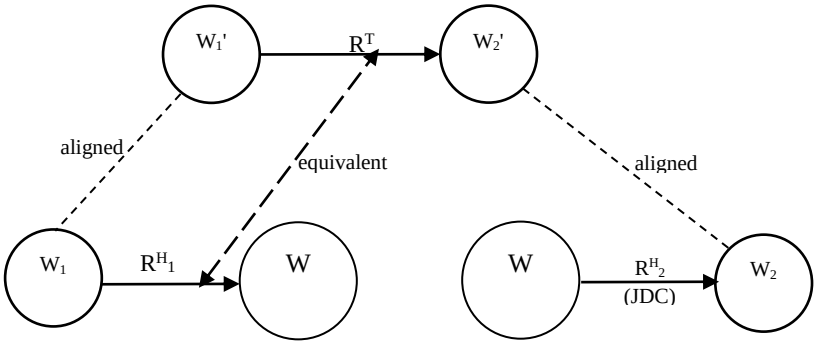

Figure. 21. Depicting Reverse of Category 1 JTD

Table 24: Examples satisfying the reverse of category 1 JTD

R^T	R^H_1	R^H_2(JDC)	T–H pairs	Dependency Triplets
agent	agent	nn	T: Miller died Thursday night, of congestive heart failure, at his home in Roxbury, Conn., surrounded by his family, his assistant, Julia Bolus, said Friday. H: Miller died Thursday night, of heart failure, surrounded by family members.	agent(surrounded-19, family-22) agent(surrounded-10, members-13) nn(members-13, family-12)
nsubj	prep_in	nn	T: Lebanon is in a period of mourning after a massive car bomb killed the former Prime Minister, Rafik Hariri, on Tuesday. H: Mr. Hariri was killed in a bomb explosion.	nsubj(killed-13, bomb-12) prep_in(killed-4, explosion-8) nn(explosion-8, bomb-7)
dobj	nsubjpass	num	T: On 12 August, the San Carlos Battalion came across mines placed in their path and one soldier was killed while two were seriously injured. Meanwhile on 10 August, urban commandos took a patrol car by surprise and dropped a grenade inside the car, injuring four and partially destroying the vehicle. H: Four people were injured by a grenade.	dobj(injuring-22, four-23) nsubjpass(injured-4, people-2) num(people-2, Four-1)

- **Negation handling on JTD:** For each of the six categories of JTD presented earlier, negation relation is handled with a different scoring mechanism. Table 25 presents an example T–H pair from the RTE1 devset which belongs to category 1 of JTD and having a negation relation associated with the text. If the text token W is associated with a negation relation, then the neg_bit value of the governor node W_1 of the hypothesis triplet $R^H(W_1, W_2)$ is set to 1.

Table 25: Example of negation for category 1 of JTD

R^H	R^T_1	R^T_2 (JDC)	T–H pair	Dependency Triplets
nsubj	nsubjpass	prep_of	T: **No stockpiles** of weapons of mass destruction have been found in Iraq since Saddam's regime was toppled in a US-led invasion last year. H: Weapons of mass destruction found in Iraq.	nsubjpass(found-10, stockpiles-2) prep_of(stockpiles-2, weapons-4) det(stockpiles-2,**No**-1) nsubj(found-5, Weapons-1)

3.7.2 Semantic Similarity Matching Module: If any hypothesis triplet $R^H(W_1,W_2)$ cannot be inferred from any text triplet(s) either jointly (JTD) or individually (STD), this semantic similarity matching module is invoked. This semantic similarity module generates a semantic similarity score for any given pair of words. WordNet::Similarity[6] package is used to achieve this purpose for the present study. Eight different similarity metrics are available in the WordNet::Similarity package - HirstStOnge, LeacockChodorow, Lesk, WuPalmer, Resnik, JiangConrath, Lin and Path based. For the present study, we considered the WuPalmer metric as the semantic similarity measure.

Two categories were considered under the semantic similarity module which are described below along with suitable diagrams.

- **Category 1**. For a particular hypothesis triplet $R^H(W_1,W_2)$, all the text triplets $R^T(W_S',W_2')$ are identified such that the tokens W_2–W_2' are lexically aligned to each other by any WordNet relations other than antonyms and the relation pair R^T–R^H are equivalent according to matching rule 2 of STD. Successively, the semantic similarity module is invoked to generate the semantic similarity scores between the hypothesis

[6] http://wn-similarity.sourceforge.net

token W_1 and the text token W_S' associated with the identified text triplets $R^T(W_S',W_2')$. The highest among these semantic similarity scores is stored in a variable, say S_1. Figure. 22.a depicts this case pictorially.

Next, the text triplets in the form of $R^T(W_2',W_S')$ are also identified where W_2–W_2' are lexically aligned by other than the antonym relation and the relation pair R^T–R^H are equivalent as per matching rule 3 of STD. Again the semantic similarity module is invoked to compute the semantic similarity scores between all pairs of the tokens W_S'–W_1 of the identified text triplets and the maximum of these scores is stored in another variable, say S_2. This case is illustrated in Figure. 22.b**.**

Finally the greater among the two scores, score1 and score2 is considered and assigned in the *P* vector of the child node W_2 of the particular branch $R^H(W_1,W_2)$ in the hypothesis dependency tree.

- **Category 2.** This category determines all possible text triplets $R^T(W_1',W_S')$ for a particular hypothesis triplet $R^H(W_1,W_2)$ where the tokens W_1–W_1' are lexically aligned by any WordNet relations other than antonyms and the relations R^T–R^H are equivalent according to matching rule 2 of STD. Semantic similarity is computed for all pair of words W_2–W_S' and the highest score is stored in S_1. Figure. 23.a diagrammatically depicts this case.

  Next, the text triplets of the form $R^T(W_S',W_1')$ are identified where R^T-R^H pair is equivalent as per matching rule 3 of STD and W_1–W_1' are lexically aligned. Then the semantic similarity scores between the token pairs W_2–W_S' are calculated and the highest among them is stored in S_2. The diagrammatic representation is shown in Figure. 23.b.

  The greater among S_1 and S_2 is assigned to the node W_2 of the hypothesis dependency graph.

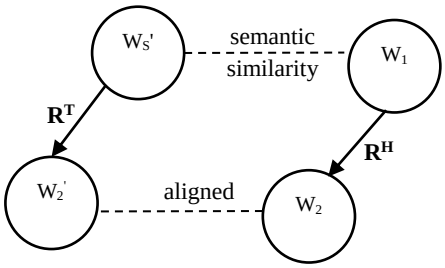

Figure. 22. a. Case 1 of Category 1

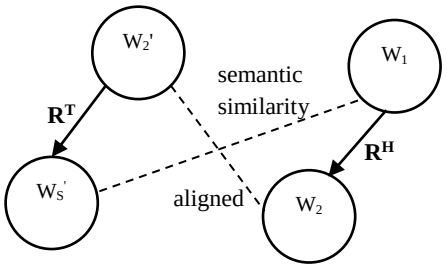

Figure. 22. b. Case 2 of Category 1

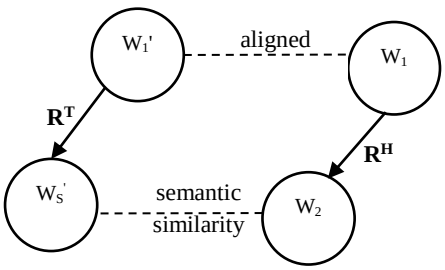

Figure. 23. a. Case 1 of Category 2

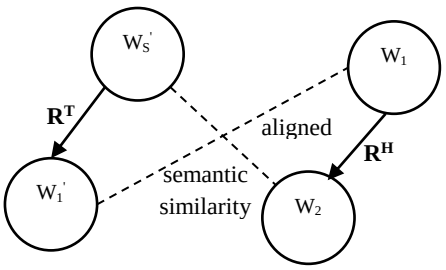

Figure. 23. b. Case 2 of Category 2

However, we noticed that the Wu-Palmer metric generates semantic similarity scores greater than 1 for some word pairs. For example, it produces a score of 1.333 for the word pair *scientist-patient*. For such cases the semantic similarity score is rounded off to 1 before it is assigned to a node. Unless the semantic similarity scores are upper bounded by 1, the final entailment score between a T–H pair may exceed the upper bound of 1.

Let us consider the following example T–H pair from the RTE1 development set.

Example 9:

<pair id="154" value="TRUE" task="RC">
<t>A male gorilla escaped from his cage in the Berlin zoo and sent terrified visitors running for cover, the zoo said yesterday.</t>
<h>A gorilla escaped from his cage in a zoo in Germany.</h>
</pair>

Table 26 lists the dependency triplets for the above T–H pair in Example 9. Only a few of the triplets are presented due to space constraints.

Table 26: Dependency triplets produced by Stanford dependency parser for T–H pair in Example 9

Text triplets	Hypothesis triplets
T1: det(gorilla-3, A-1)	H1: det(gorilla-2, A-1)
T2: amod(gorilla-3, male-2)	H2: nsubj(escaped-3, gorilla-2)
T3: nsubj(escaped-4, gorilla-3)	H3: root(ROOT-0, escaped-3)
T4: poss(cage-7, his-6)	H4: poss(cage-6, his-5)
T5: prep_from(escaped-4, cage-7)	H5: prep_from(escaped-3, cage-6)
T6: det(zoo-11, the-9)	H6: det(zoo-9, a-8)
T7: nn(zoo-11, Berlin-10)	H7: prep_in(escaped-3, zoo-9)
T8: prep_in(escaped-4, zoo-11)	**H8: prep_in(zoo-9, Germany-11)**
T9: root(ROOT-0, said-22)	

From the above table it can be noted that the hypothesis triplets H1, H2, H4, H5 and H7 directly match with the text triplets T1, T3, T4, T5 and T8, respectively following matching rule 1. Triplet H6 follows the STD matching rule 7. Only H8 do not directly match with any of the generated text triplets. Therefore, the semantic similarity module tries to find out the set of all relations R^T of the text which are equivalent to *prep_in* (R^H). In triplet T7, the relation *nn* is equivalent to *prep_in* according to matching rule 2 of the STD category and the corresponding hypothesis governor word *zoo* directly matches with the corresponding text governor word. However, the corresponding dependent words (i.e., *Berlin* and *Germany*) do not match. Therefore, semantic similarity score is computed for the word pair *Berlin* and *Germany*. The WordNet::Similarity package implementation of the WuPalmer metric returns a score of 0.5 for the given word pair which is then assigned to the token *Germany* of the hypothesis dependency graph.

A few more examples from the different RTE development sets are presented in Table 27 and their corresponding dependency triplets, only the relevant ones with respect to semantic similarity, are provided in Table 28 to depict how the semantic similarity scores are assigned to dependency triplet pairs.

Apart from the WordNet::Similarity package, we also explored another semantic similarity measuring tool gensim[7] which is based on Word2Vec [33], a distributional semantics model based on vector representation of words (i.e., *word embeddings*). The main working principle behind Word2vec is that words appearing in the similar context have similar meaning. The Word2vec model takes a large text corpus as input and produces a vector space as output with each distinct word in the corpus being represented as a vector in that space. The similarity between a word pair is calculated by measuring the cosine of the angle between their corresponding word vectors. The lower the angle between the two vectors, the more similar the two words are. For our experimental purpose, we have used 1.5 GB GoogleNews corpus as the underlying corpus on which the models of Word2vec are trained. Word2Vec contains two distinct models – the Continuous Bag-of-words (CBOW) and skip-gram, each with two different training methods. The CBOW model predicts the target words from the source context words and the skip-gram model predicts the source context words from the target words. However, due to low scores produced by this tool, we did not embed it in our method as the optimal performance was achieved by the WordNet::Similarity package.

[7] https://radimrehurek.com/gensim/

Table 27: Example T–H pairs relevant for semantic similarity matching module

Pair Id #	Dataset	T–H pairs
1	RTE1 dev1	T: Guerrillas killed a peasant in the city of Flores H: Guerrillas killed a civilian.
2	RTE1 dev1	T: Swedish massage is used to help relax muscles, increase circulation, remove metabolic waste products, help the recipient obtain a feeling of connectedness, a better awareness of their body and the way they use and position it. H: Swedish massage loosens tense muscles.
3	RTE3 dev	T: Airbus could site a design engineering centre in the Midlands region of the UK to take advantage of the availability of skilled engineering staff following the demise of MG Rover, the collapsed UK carmaker. H: Airbus plans a design engineering centre.

Table 28: Dependency triplets relevant to semantic similarity for the T–H pairs in Table 27

Paid Id #	Text dependency triplets (equivalent to the hypothesis triplets)	Hypothesis triplets (for which semantic similarity module is invoked)
1	dobj(killed-2, peasant-4) prep_in(killed-2, city-7)	dobj(killed-2, civilian-4)
2	amod(massage-2,Swedish-1), nsubjpass(used-4, massage-2) dobj(relax-7, muscles-8)	nsubj(loosens-3, massage-2) amod(muscles-5, tense-4)
3	nsubj(site-3, Airbus-1)	nsubj(plans-2, Airbus-1)

For the three T–H pairs in Table 27, the semantic similarity scores between the concerned tokens and the final score assigned to the nodes of the hypothesis graphs are depicted in the Table 29. To compare the two semantic similarity models and their effects on the textual entailment decision, scores generated by both the gensim tool and the WordNet::Similarity package based Wu-Palmer measure are presented in the table. It is evident from the table that the gensim scores are much lower than the ones produced by WordNet::Similarity package. Therefore for optimal performance we have only considered the Wu-Palmer scores.

Table 29: Illustrating semantic score calculation

Pair Id#	Equivalent R^H-R^T pair	Equivalent according to STD matching rule #	Token pair for semantic score calculation	gensim score	Wu-Palmer score	Final score assigned	Score assigned on node
1	dobj-dobj	1	peasant-civilian	0.1981613	0.727273	0.727273	civilian
	dobj-prep_in	2	city-civilian	0.1902787	0.533333		
2	nsubj- amod	3	loosen-swedish	0.01978217	0.0	0.4	massage
	nsubj-nsubjpass	2	loosen-use	0.22086817	0.4		
	amod-dobj	3	tense-relax	0.21001716	0.5	0.5	tense
3	nsubj-nsubj	1	plan-site	0.20739367	0.285714	0.285714	Airbus

3.8 Entailment score calculation

As mentioned earlier, each node in the hypothesis dependency graph maintains 2 bits: antonym bit (ant_bit) and negation bit (neg_bit) and four scoring components listed as follows.

- The predecessor_score vector (*P*)
- The average of predecessor_score vector (*A*)
- The child_score (*C*)
- The total_score (*T*)

Since the dependency triplets when combined together form a graph with some nodes having more than one predecessor, the *P* vector component of each node is represented as a vector rather than a single variable. The dependency graph comparison module (cf. Section 3.7) assigns a matching score in the range of [-1, 1] to the *P* vector component of the nodes in the hypothesis graph. Each component of the *P* vector of a particular node represents the similarity score between each of its predecessor and its corresponding aligned token in the text. If a hypothesis–text token pair is lexically aligned to each other by any WordNet relation, a complete matching score of 1 is assigned. However, if the token pair is aligned by antonym relation, a score of -1 is assigned to indicate contradiction. Finally a semantic similarity score is calculated between the corresponding token pair if they cannot be aligned to each other by any of the given WordNet relations provided in Table 5. As a node can have multiple predecessors, the values of the *P* vector are averaged and assigned to the score component *A*. Like predecessors, a node can have multiple successors or children as well. The *C* score component of a node is calculated by summing up the *T* score of all its children and then normalizing it by the number of children. As in a graph there are only 2 constituents of a node: the predecessors and the successors, finally the *T* score component of the node is calculated by taking the average of the values of *A* score and *C* score. This *T* score indicates the sum total or the effective score of a node. Since the final entailment score obtained at the end of the graph traversal process is upper bounded by the value of 1, each of the score components of *A*, *C* and *T* of the nodes in the graph are normalized in each step to lie below 1.

After scores have been assigned to each node, the hypothesis dependency graph is traversed in level order fashion from the bottommost level to the topmost level. During this process of traversal, the scores of the nodes belonging to a particular level are accumulated and gradually propagated to the next higher levels till it reaches the root node at the topmost level. Table 30 illustrates how the score components of the several categories of nodes in the dependency graph are computed. The topmost node in the graph which is connected to the dummy ROOT node is assigned a value of 0 in its *A* score since it has no valid predecessor. Therefore the *T* score of this node is directly set to the value of its *C* score. Since the leaf nodes have no branch or children, their *C* score is set to 0 and consequently their *T* score is assigned to the value of their *A* score directly. For each non-leaf node in the graph, the *C* score is calculated by taking the average of the *T* score of all its children and the *T* score component is set to the average of its *C* score and *A* score values. The scores of the nodes in a graph are computed and propagated in a bottom-up fashion till the final score for the entire graph is obtained at the ROOT node. This final score is considered as the entailment score for the given T–H pair. This score is then fed to the subsequent module which takes the final decision about whether it is a case of YES or NO entailment.

Table 30: Scoring mechanism for the various nodes in the hypothesis dependency graph

Types of Nodes	***P* score vector**	***A* score**	***C* score**	***T* score**
Leaf nodes	Scores assigned in the matching module	Average of all the P score vector components	0	A score
Non-leaf non-root nodes	Scores assigned in the matching module	Average of all the P score vector components	Average of T score of all its children	$\frac{1}{2}$(A score+C score)
Topmost node immediate to ROOT	0	0	Average of T score of all its children	C score

However, the scoring mechanism as depicted in Table 30 is applicable only to the nodes which have their ant_bit and neg_bit values both set to 0. If any of the two bits for a particular node is set to 1, then the above scoring mechanism is not applied to them. During the traversal of the hypothesis dependency graph from the

bottommost level to the top level, if any node is encountered with either of their ant_bit or neg_bit value set to 1, the *T* score component of that node is directly set to a value of -1 without going into the above process of calculation of the score components. The motivation behind this is to neutralize the effect of the computed score of its children nodes traversed so far.

An example T–H pair from the RTE1-dev1 set is provided in Example 10 and the corresponding graph structure of the hypothesis is presented in Figure. 24 to illustrate the matching process and the scoring mechanism.

Example 10:

T: The report catalogues 10 missed opportunities within the CIA and FBI to uncover pieces of the September 11 plot.
H: Ten missed opportunities within the CIA and FBI are uncovered in the report.

Table 31 presents the dependency triplets for the T–H pair in Example 10. The matching of the hypothesis triplets with the text triplets according to the comparison module (cf. Section 3.7) is presented in Table 32.

Table 31: Dependency triplets of the T–H pair in Example 10

Text Dependency Triplets	Hypothesis Dependency Triplets
T1: det(report-2, The-1)	
T2: nsubj(catalogues-3, report-2)	
T3: root(ROOT-0, catalogues-3)	**H1**: num(opportunities-3, Ten-1)
T4: num(opportunities-6, 10-4)	**H2**:amod(opportunities-3,missed-2)
T5: amod(opportunities-6, missed-5)	**H3**:nsubjpass(uncovered-10,opportunities-3)
T6: dobj(catalogues-3, opportunities-6)	**H4:** det(CIA-6, the-5)
T7: det(CIA-9, the-8)	**H5**:prep_within(opportunities-3,CIA-6)
T8:prep_within(opportunities-6,CIA-9)	**H6**:prep_within(opportunities-3, FBI-8)
T9: prep_within(opportunities-6, FBI-11)	**H7**: conj_and(CIA-6, FBI-8)
T10: conj_and(CIA-9, FBI-11)	**H8**: auxpass(uncovered-10,are-9)
T11:aux(uncover-13, to-12)	**H9**: root(ROOT-0, uncovered-10)
T12: infmod(opportunities-6, uncover-13)	**H10**: det(report-13, the-12)
T13: dobj(uncover-13, pieces-14)	**H11**: prep_in(uncovered-10, report-13)
T14: det(plot-19, the-16)	
T15: nn(plot-19, September-17)	
T16: num(plot-19, 11-18)	
T17: prep_of(pieces-14, plot-19)	

Table 32: Matching between dependency triplets of the T–H pair of Example 10 according to the matching rules

Hypothesis Dependency Triplet #	Lexical Match with				Semantic Match with			
	Text Dependency Triplet#	Satisfying matching rule#	Assigned Score	P score assigned on node	Text Dependency Triplet#	Word pair for semantic similarity measurement	Assigned semantic score (WuPalmer)	P score assigned on node
H1	T4	1	1.0	Ten	–	–	–	–
H2	T5	1	1.0	Missed	–	–	–	–
H3	T12	3	1.0	Opportunity	–	–	–	–
H4	T7	1	1.0	The	–	–	–	–
H5	T8	1	1.0	CIA	–	–	–	–
H6	T9	1	1.0	FBI	–	–	–	–
H7	T10	1	1.0	FBI	–	–	–	–
H8	–	7	1.0	Are	–	–	–	–

H9	–	–	0	Uncover	–	–	–	–
H10	T1	1	1.0	The	–	–	–	–
H11	–	–	–	–	T2	catalogue-uncover	0.2222	report

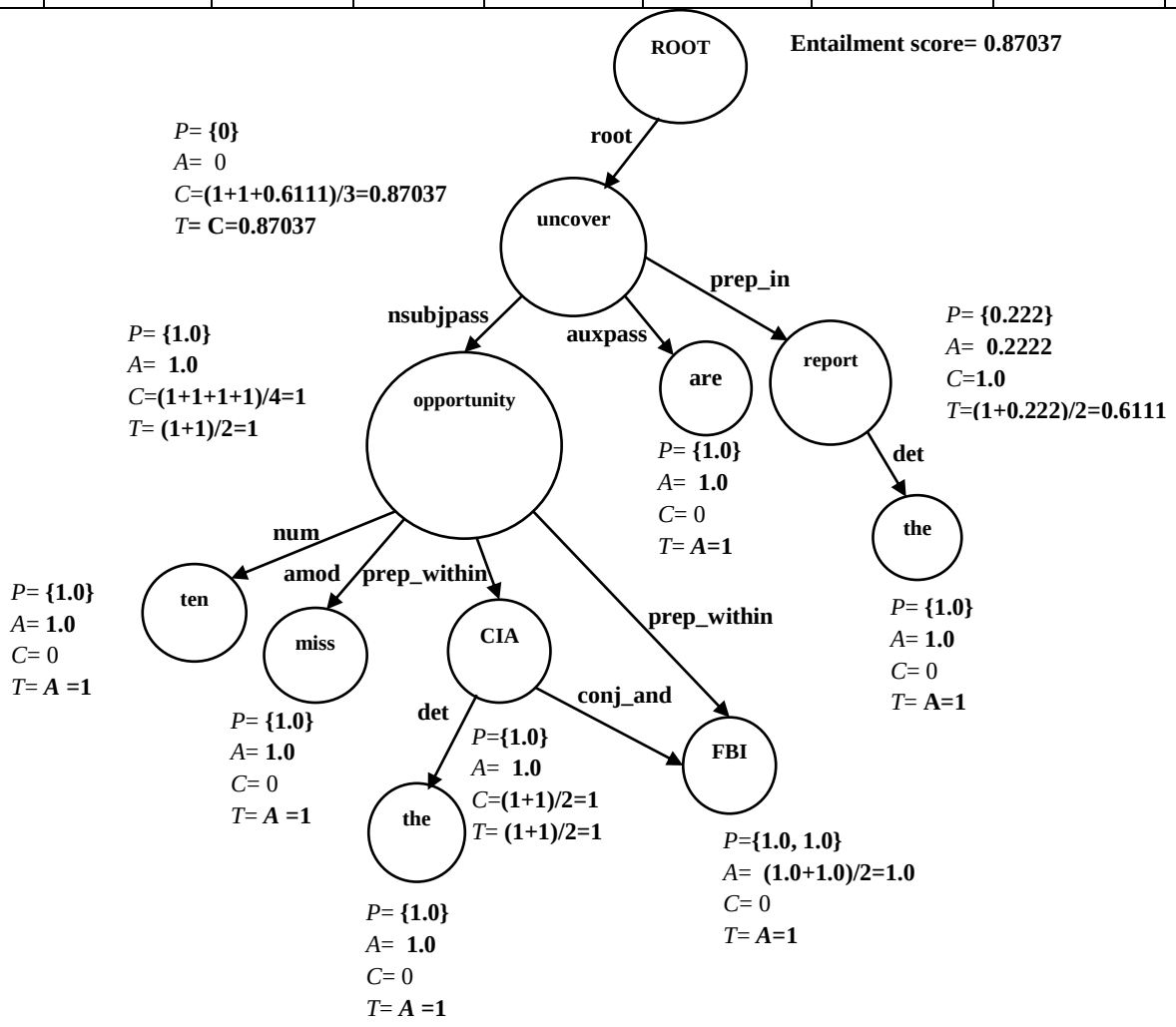

Figure. 24. Component scores of the dependency graph

The various score components of each node in the hypothesis dependency graph are shown Table 33.

Table 33: Different score components of the hypothesis graph in Figure. 24

Node	***P* score vector**	***A* score**	***C* score**	***T* score**
Ten	{1.0}	1.0	0	1
Missed	{1.0}	1.0	0	1
opportunity	{1.0}	1.0	1	1
The	{1.0}	1.0	0	1
CIA	{1.0}	1.0	1	1
FBI	{1.0,1.0}	1.0	0	1
Are	{1.0}	1.0	0	1
uncover	{0}	0	0.87037	0.87037
The	{1.0}	{1.0}	0	1
Report	{0.222}	0.222	1.0	0.6111

To illustrate the Rule 4 for handling antonyms, let us consider the T–H pair of Example 10 with a slight modification as follows:

Example 11:

T: The report catalogues 10 missed opportunities within the CIA and FBI to **cover** pieces of the September 11 plot.
H: Ten missed opportunities within the CIA and FBI are uncovered in the report.

The word '*uncover*' in the text of the previous T–H pair has been changed to its antonym, '*cover',* as highlighted in the above text. Table 34 shows the various score components of the nodes in the hypothesis dependency graph.

Table 34: Different score components of the hypothesis graph for Example 11

Node	***P* score vector**	***A* score**	***C* score**	***T* score**
Ten	{1.0}	1.0	0	1
Missed	{1.0}	1.0	0	1
opportunity	**{-1.0}**	**-1.0**	1	**0**
The	{1.0}	1.0	0	1
CIA	{1.0}	1.0	1	1
FBI	{1.0,1.0}	1.0	0	1
Are	{1.0}	1.0	0	1
Cover	{0}	0	0.537	0.537
The	{1.0}	{1.0}	0	1
Report	{0.222}	0.222	1.0	0.6111

3.9 Entailment Decision The entailment score obtained from the previous step for a particular T–H pair is passed into this module. This score is then compared with various predetermined threshold values to take the final entailment decision. A score below the threshold indicates a NO entailment and a score above the threshold indicates a YES entailment. The thresholds were trained on various development datasets. The threshold for which the method results in the maximum accuracy for a development set is then applied on the corresponding RTE testsets.

4. Datasets

We performed our experiments on the RTE datasets from RTE1 to RTE4. The development set of RTE1 is divided into 2 parts. Devset 1 and devset 2 consist of 287 and 280 T–H pairs, respectively, divided into 7 tasks as Comparable Documents (CD), Information Extraction (IE), Information Retrieval (IR), Machine Translation (MT), Question Answering (QA), Reading Comprehension (RC) and Paraphrase Acquisition (PP). RTE1 testset contains 800 T–H pairs divided into the same 7 tasks as before. The RTE2 and RTE3 datasets contain 1600 T–H pairs each equally divided into a development set and a testset. The RTE4 dataset does not contain any development set; only a testset is provided with 1000 T–H pairs. The T–H pairs of RTE2, RTE3 and RTE4 datasets are divided into 4 tasks of IE, IR, QA and Summarization (SUM). The T–H pairs of the RTE1, RTE2 and RTE3 datasets are all classified for the 2 way entailment task; i.e they are labelled either as cases of YES entailment or NO entailment. However, the T–H pairs of the RTE4 dataset are marked for the 3 way classification task as ENTAILMENT, CONTRADICTION and UNKNOWN. Together CONTRADICTION and UNKNOWN classes are grouped under the case of NO entailments in the gold standard output of the RTE4 testset. Table 35 presents the statistics of the development and test sets of each of the datasets used here for experimental purpose.

Table 35. Statistics of the RTE datasets

	Dataset	Taskwise division								
		IE	**IR**	**QA**	**SUM**	**CD**	**MT**	**PP**	**RC**	**Total**
RTE 1	dev1	20	50	50	–	50	8	58	51	287
	dev2	50	20	40	–	48	46	24	52	280
	test	120	90	130	–	150	120	50	140	800
RTE2	dev	200	200	200	200	–	–	–	–	800
	test	200	200	200	200	–	–	–	–	800
RTE3	dev	200	200	200	200	–	–	–	–	800
	test	200	200	200	200	–	–	–	–	800
RTE4	test	300	300	200	200	–	–	–	–	1000

5. Experiments and Results

This section presents the results obtained from the experiments performed on various datasets ranging from RTE1 to RTE4. The thresholds were trained on 4 development sets, 2 devsets of RTE1 dataset and 1 devset each for RTE2 and RTE3 dataset. For each development set, learning curve experiments were carried out using varying thresholds for each task. The threshold that produces the optimum result on a particular development set is used to evaluate the performance of the method on the corresponding testset.

The method was evaluated in terms of accuracy which is defined as equation (1), where TP and TN refer to true positives and true negatives, respectively.

$$\boldsymbol{accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{Total\ number\ of\ T-H\ pairs}} * \mathbf{100} \qquad \textbf{(1)}$$

Results obtained by the learning curve experiments on RTE1 dev1 set and RTE1 dev2 set are presented in Tables 36 and 37 respectively. Task specific highest scores are shown in bold in Tables 36 and 37. As can be observed from Table 36, task–wise highest accuracy varies between **60%** to **70%** . The method attains the highest accuracy for the tasks of CD and IE while the lowest accuracy is produced for the task of IR. The highest value of the overall accuracy for the RTE1 dev1 set is **62.02%** obtained for two thresholds, 0.6 and 0.7.

Table 36: Accuracy of the method for the different tasks for different thresholds on RTE1- dev1 set

Task	**Thresholds**														
	0.25	**0.3**	**0.35**	**0.4**	**0.45**	**0.5**	**0.55**	**0.6**	**0.65**	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**
CD	60	60	60	68	**70**	66	66	66	66	62	58	58	64	62	62
IE	60	60	60	55	55	55	60	60	55	65	**70**	65	65	65	60
IR	52	48	50	48	50	54	54	58	58	**60**	**60**	58	56	58	58
MT	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	**62.5**	50	37.5	**62.5**	50
QA	48	52	52	54	58	64	64	**66**	62	64	60	64	64	62	58

RC	58.82	**60.78**	58.82	**60.78**	56.86	56.86	58.82	**60.78**	54.90	58.82	**60.78**	54.90	52.94	56.86	56.86
PP	53.45	55.17	60.34	58.62	55.17	55.17	60.34	60.34	**63.79**	**63.79**	62.07	**63.79**	60.34	62.07	56.89
Overall	55.05	55.75	56.79	57.84	57.84	58.89	60.63	**62.02**	60.63	**62.02**	60.98	59.93	59.23	60.63	58.19

Table 37: Accuracy of the method for the different tasks for different thresholds on RTE1- dev2 set

Task	**Thresholds**														
	0.25	**0.3**	**0.35**	**0.4**	**0.45**	**0.5**	**0.55**	**0.6**	**0.65**	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**
CD	56.25	58.33	58.33	60.41	54.16	56.25	58.33	60.41	**64.58**	58.33	58.33	58.33	58.33	60.41	58.33
IE	54	54	54	54	54	54	52	50	48	52	54	**56**	50	**56**	54
IR	**65**	60	55	55	50	50	45	40	40	40	45	50	55	55	55
MT	52.17	50	50	47.83	52.17	50	50	**54.35**	50	45.65	47.83	47.83	45.65	45.65	47.83
QA	**47.5**	45	45	45	40	32.5	37.5	35	35	35	37.5	37.5	32.5	35	37.5
RC	51.92	53.85	53.85	53.85	55.77	53.85	53.85	57.69	51.92	57.69	50	50	51.92	**59.62**	**59.62**
PP	50	50	50	50	50	50	58.33	58.33	58.33	54.17	62.5	62.5	**66.67**	62.5	62.5
Overall	**53.21**	52.86	52.5	52.5	51.43	50	51.07	51.79	50.36	50	50.71	51.43	50.71	52.86	**53.21**

Table 37 shows that task–wise highest accuracy varies from 47.5% to 66.67% with the highest accuracy obtained for the PP task and the lowest accuracy for the QA task. The overall accuracy for this devset reaches the highest value of **53.21%** for two threshold values of 0.25 and 0.95.

Figure. 26 and 27 show the results of the learning curve experiments for each of the tasks on the RTE1 dev1 and RTE1 dev2 datasets, respectively. The x–axis of the graph represents the threshold values and the system accuracies (in percentage) are plotted along the y–axis .

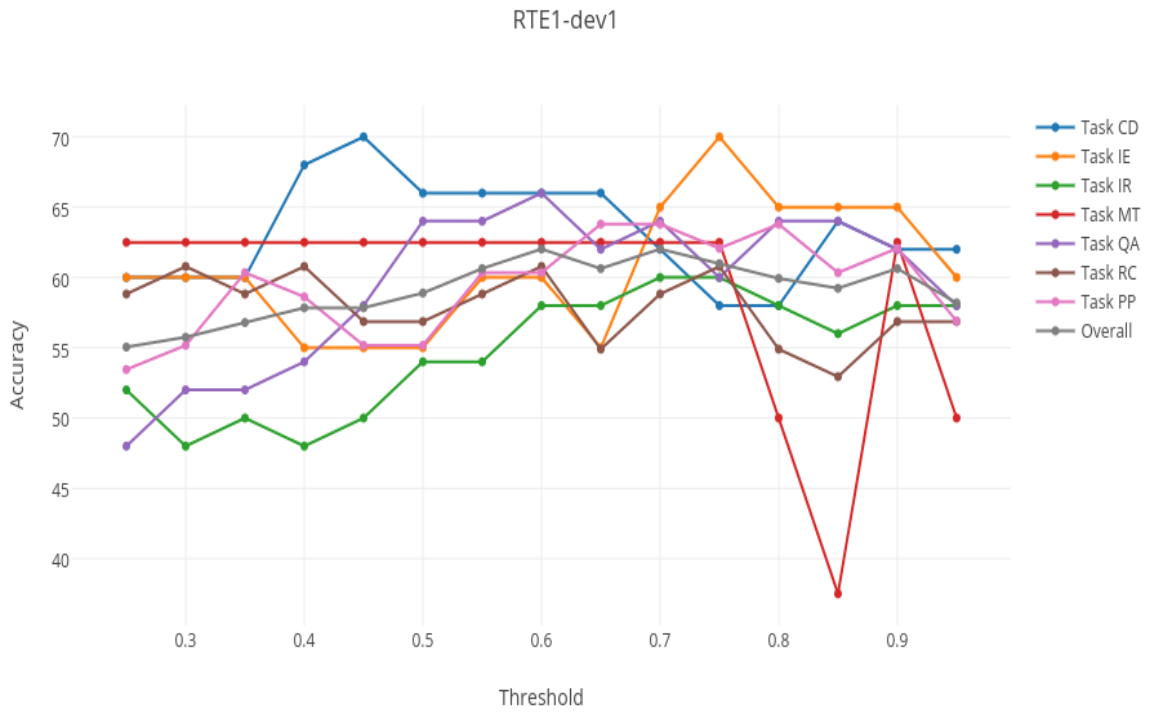

Figure. 26. Results of learning curve experiments on the RTE1 dev1 set

Figure. 26 shows that the accuracy reaches the peak value for the tasks of CD and IE. However, the curve for the task MT drops abruptly at threshold 0.85. There are only 8 T–H pairs belonging to the MT task in the RTE1–dev1 set, thus one pair accounts for 12.5% accuracy. At threshold 0.75, only 3 and 2 T–H pairs are identified as true positive and true negative, respectively, thus generating an accuracy of 62.5%. A change in the threshold from 0.75 to 0.8 causes the number of true positive pairs to decrease from 3 to 2 resulting in a high fluctuation in accuracy from 62.5% to 50%. Again at threshold 0.85, the number of true positives decreases by 1 causing a further drop of 12.5% in accuracy. The number of true negative pairs remains fixed in this threshold range. For threshold 0.9, the number of true negative pairs increases from 2 to 4 thereby producing a sudden raise in accuracy to 62.5% again. Therefore, it is clear that this curve exhibits abrupt rise and fall due to very small number of T–H pairs present in the dataset. A slight change in the number of true positive and/or true negative T–H pairs for successive thresholds causes the accuracy of the method to deviate in an unexpected manner.

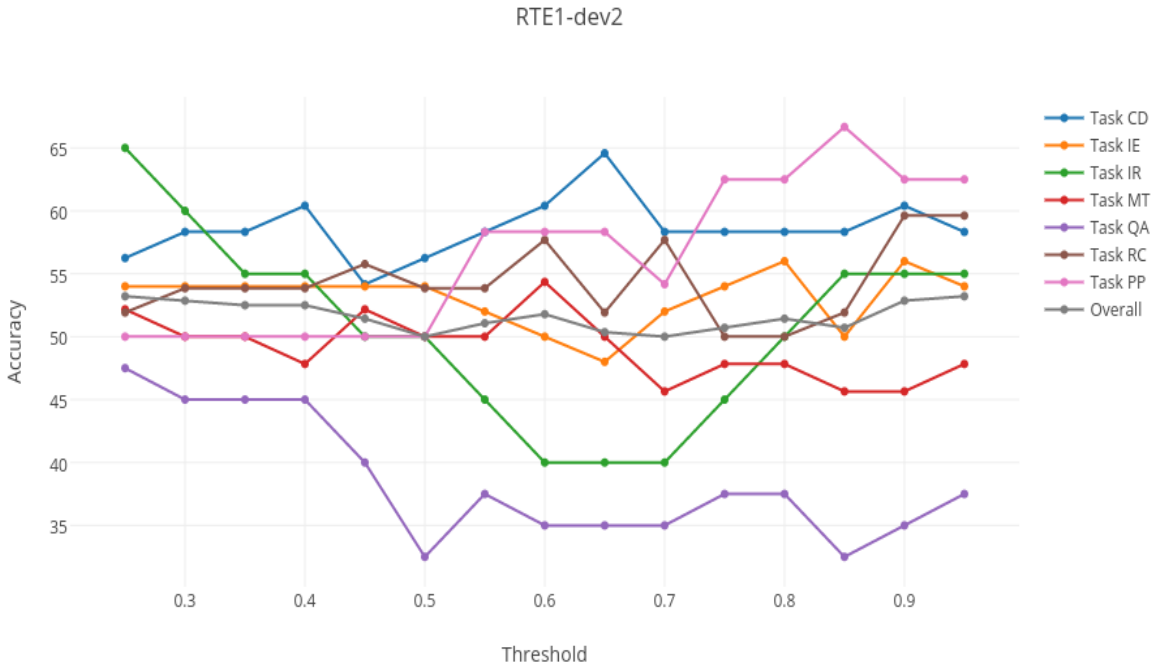

Figure. 27. Results of learning curve experiments on the RTE1 dev2 set

From the diagram Figure. 27, it is clearly observed that the method exhibits the highest and lowest accuracy for the PP and QA task respectively for the RTE1-dev2 dataset. The overall accuracy reaches the peak value for two extreme thresholds 0.25 and 0.95.

The two optimal thresholds obtained from the 2 development sets of RTE1 are then used to calculate the accuracy on the RTE1 testset. The performance of the method in terms of precision, recall and F-score for each of the 7 tasks are presented in Table 38. Although the accuracy of the method for the testset is measured on the basis of the optimal thresholds learned from the development set, it was observed that the accuracy attains the highest value for some thresholds which are different from the optimal threshold values obtained from the devsets. Table 38, in addition, shows the optimal results obtained by the proposed method on the RTE1 testset for each of the tasks and the corresponding threshold values. It was observed that the method performs the best for the task of CD and the worst performance is produced for the task of QA. Moreover, for the task of MT the method shows the best accuracy for two different thresholds, 0.25 and 0.85. The overall accuracy of the method for this testset attains a value of **60.5%.**

Table 38: Evaluation results on the RTE1 testset

Task	**Devset**	**Optimum threshold**	**Positive TE**			**Negative TE**			**Overall Accuracy**
			Pr	**Re**	**F**	**Pr**	**Re**	**F**	
CD	RTE1 dev1	0.45	0.65	0.893	0.752	0.829	0.52	0.639	0.706
	RTE1 dev2	**0.65**	0.831	0.72	0.772	0.753	0.853	0.799	**0.787**
	–	***0.65***	*0.831*	*0.72*	*0.772*	*0.753*	*0.853*	*0.799*	***0.787***
IE	RTE1 dev1	0.75	0.519	0.467	0.492	0.515	0.566	0.539	0.516
	RTE1 dev2	0.8	0.548	0.383	0.451	0.526	0.683	0.594	0.533
	–	**0.85**	0.613	0.317	0.418	0.539	0.8	0.644	**0.558**
IR	RTE1 dev1	0.7	0.5	0.356	0.416	0.5	0.64	0.561	0.5

	RTE1 dev2	0.25	0.526	0.91	0.667	0.667	0.178	0.281	0.544
	–	**0.3**	0.533	0.889	0.666	0.667	0.22	0.331	**0.556**
MT	RTE1 dev1	**0.25**	0.525	0.88	0.658	0.632	0.2	0.304	**0.542**
	RTE1 dev2	0.6	0.49	0.417	0.451	0.493	0.567	0.527	0.492
	–	**0.85**	0.692	0.15	0.247	0.523	0.933	0.67	**0.542**
QA	RTE1 dev1	0.6	0.435	0.462	0.448	0.426	0.4	0.413	0.431
	RTE1 dev2	0.25	0.496	0.862	0.629	0.471	0.123	0.195	0.492
	–	**0.3**	0.5	0.846	0.629	0.5	0.154	0.235	**0.5**
RC	RTE1 dev1	0.3	0.511	0.957	0.666	0.667	0.086	0.152	0.521
	RTE1 dev2	**0.95**	0.758	0.357	0.485	0.579	0.886	0.7	**0.621**
	–	***0.95***	*0.758*	*0.357*	*0.485*	*0.579*	*0.886*	*0.7*	***0.621***
PP	RTE1 dev1	**0.7**	0.652	0.6	0.625	0.629	0.68	0.654	**0.64**
	RTE1 dev2	0.85	0.538	0.28	0.368	0.514	0.76	0.613	0.52
	–	***0.7***	*0.652*	*0.6*	*0.625*	*0.629*	*0.68*	*0.654*	***0.64***
Overall	RTE1-dev1	–	0.537	0.689	0.604	0.669	0.409	0.508	0.549
	RTE1-dev2	–	0.584	0.576	0.58	0.584	0.591	0.588	0.584
	–	–	0.619	0.541	0.578	0.594	0.668	0.629	**0.605**

Comparison of our experimental results with the submitted systems in the First PASCAL Recognizing Textual Entailment Challenge[8] (RTE 1) [35] shows that among the 28 submissions, only 2 works presented in [36] and [37] surpass the accuracy of our proposed method by attaining overall accuracy values of 60.6% and 70% respectively.

Table 39 presents the results obtained for different threshold values on the RTE2 development set for each of the tasks. It can be observed from the table that the method exhibits the highest accuracy for the SUM task attaining an accuracy of **75%** and the lowest score is obtained for the IE task with an accuracy of **57.5%**. It was observed that the overall accuracy reaches its peak value of **63.5%** for the threshold value 0.65. The maximum accuracies for each of the four tasks are highlighted in bold fonts in Table 39. The threshold vs. accuracy graph for the RTE2 devset is presented in Figure. 28.

Table 39: Accuracy of the method for the different tasks for different thresholds on RTE2- dev set

Task	**Thresholds**														
	0.25	**0.3**	**0.35**	**0.4**	**0.45**	**0.5**	**0.55**	**0.6**	**0.65**	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**
IE	50.5	51.5	50.5	53	53	53	56	**57.5**	53.5	51.5	47.5	45.5	50	50	51
IR	56.5	56.5	56	57.5	57.5	57.5	57.5	**60**	59.5	59	57	59	58.5	59	57.5
QA	55	55	55.5	56	58	61	63.5	64.5	66	66.5	69.5	69.5	**72.5**	69	71
SUM	57.5	60	59.5	62	66.5	69	69	71.5	**75**	71	71.5	69.5	66	62	58.5
Overall	54.88	55.75	55.38	57.13	58.75	60.13	61.5	63.38	**63.5**	62	61.38	60.88	61.75	60	59.38

[8] http://u.cs.biu.ac.il/~nlp/RTE1/Proceedings/

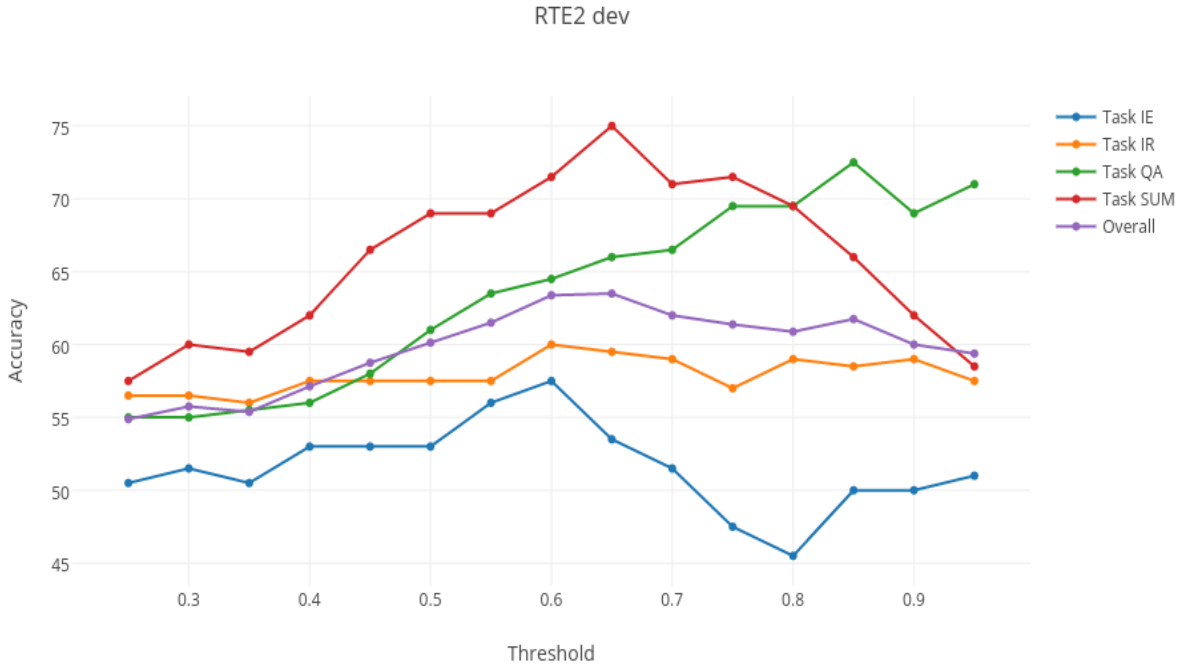

Figure. 28. Results of learning curve experiments on the RTE2 dev set

From the learning curve experiments (cf. Table 40 and Figure. 28), it is evident that the accuracy attains the highest value for the SUM task and the lowest accuracy is resulted for the IE task. The overall accuracy of the method reaches its peak value for the threshold 0.65.

The thresholds for which the method resulted in the best performance for each of the tasks on the RTE2 development set were applied on the corresponding testset tasks. Table 40 presents the performance of the method for each of the tasks in RTE2 testset in terms of precision, recall and F-score. In addition to the scores obtained with the thresholds learnt from the RTE2 devset, it also shows the optimal performances that can be obtained with the proposed method. The overall accuracy has reached the value of **64.5%** for this testset.

Table 40: Evaluation results on the RTE2 testset

Task	**Devset**	**Optimum threshold**	**Positive TE**			**Negative TE**			**Overall Accuracy**
			Pr	**Re**	**F**	**Pr**	**Re**	**F**	
IE	RTE2	0.6	0.551	0.75	0.635	0.609	0.39	0.475	0.57
	–	0.55	0.558	0.82	0.664	0.66	0.35	0.457	**0.585**
IR	RTE2	0.6	0.569	0.7	0.628	0.61	0.47	0.53	0.585
	–	0.75	0.691	0.56	0.618	0.63	0.75	0.685	**0.655**
QA	RTE2	0.85	0.634	0.64	0.637	0.636	0.63	0.633	0.635

	–	0.8	0.626	0.72	0.669	0.671	0.57	0.616	**0.645**
SUM	RTE2	0.65	0.67	0.59	0.627	0.634	0.71	0.669	0.65
	–	0.55	0.659	0.81	0.727	0.753	0.58	0.655	**0.695**
Overall	RTE2	–	0.598	0.67	0.632	0.625	0.55	0.585	0.61
	–		0.624	0.727	0.672	0.674	0.563	0.614	**0.645**

Among a total of 41 participating system submissions in the Second PASCAL Recognizing Textual Entailment Challenge[9] (RTE 2) [38], only 2 systems described in [39] and [40] exhibited better performance than our proposed method with accuracy values of 75.38% and 73.75% respectively. However, these systems made use of many rich and complex resources and techniques such as Semantic Role Labeller, Logical Inference, Paraphrasing, corpus/web–based statistics in order to solve the TE problem. Our proposed method is much lighter than those in terms of prerequisite resources.

The accuracy of the method was also checked with the RTE3 development set with different threshold values the results of which are presented in Table 41. The highest accuracies of the various tasks lie within the range of **53.5%** to **71%** with the maximum score obtained for the QA task and the minimum for the IE task. The highest overall accuracy of the method came out to be **63.38%**. Figure. 29 presents the threshold versus accuracy graph for the RTE3 devset.

Table 41: Accuracy of the method for the different tasks for different thresholds on RTE3 - dev set

Task	**Thresholds**														
	0.25	**0.3**	**0.35**	**0.4**	**0.45**	**0.5**	**0.55**	**0.6**	**0.65**	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**
IE	52.5	51.5	**53.5**	54	54	50.5	51	49	50.5	53	53	52	52	50.5	49.5
IR	50.5	51.5	55.5	57.5	58.5	57	60	59.5	65	**66.5**	65.5	67	65.5	65	63
QA	56	55.5	56	58	60	61	64.5	69	70	**71**	70	70	69.5	68.5	66.5
SUM	57.5	58	61.5	61.5	61	61.5	63.5	**64.5**	**64.5**	63	62	57	55	54.5	54.5
Overall	54.13	54.13	56.63	57.75	58.38	57.5	59.75	60.5	62.5	**63.38**	62.63	61.5	60.5	59.63	58.38

[9] http://u.cs.biu.ac.il/~nlp/RTE2/Proceedings/

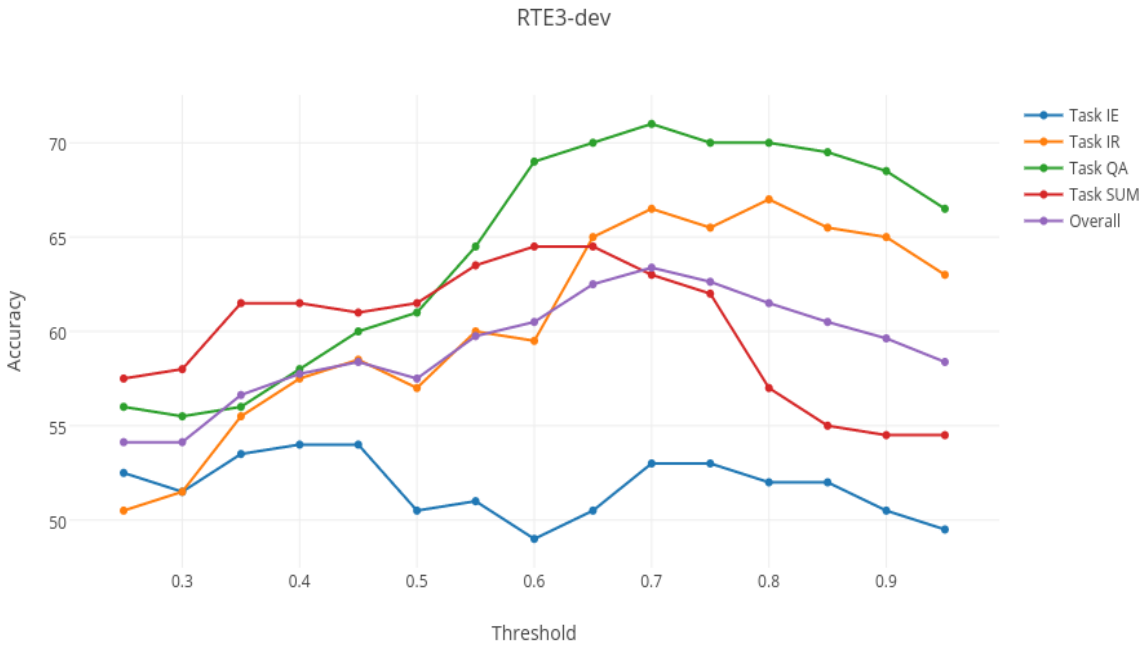

Figure. 29. Results of learning curve experiments on the RTE3 devset

Results of Learning curve experiments shown in Figure. 29 reveal that the accuracy of the method reaches the maximum value for the QA task and the minimum accuracy is produced for the IE task. At threshold 0.7, the overall accuracy attains its peak value.

The optimal thresholds for which the method produced the highest score for each of the tasks in RTE3 devset were applied on the corresponding task in the testset to evaluate the performance. Table 42 presents the performance of the method for the various tasks in terms of precision, recall and F-score. Apart from the optimal thresholds learned from the development set, the oracle performances on the testset are also provided in the table for each of the tasks. The accuracy reaches its peak value for the task QA and the overall accuracy attained by the method for this particular testset is **62.8%.**

Table 42: Evaluation Results on the RTE3 testset

Task	**Devset**	**Optimum threshold**	**Positive TE**			**Negative TE**			**Overall Accuracy**
			Pr	**Re**	**F**	**Pr**	**Re**	**F**	
IE	RTE3	0.45	0.54	0.895	0.674	0.577	0.158	0.248	0.545
	–	0.35	0.54	0.952	0.689	0.67	0.105	0.182	**0.55**
IR	RTE3	0.8	0.593	0.218	0.319	0.595	0.885	0.712	0.595
	–	0.6	0.575	0.483	0.525	0.646	0.726	0.684	**0.62**
QA	RTE3	0.7	0.748	0.755	0.751	0.72	0.713	0.716	0.735
	–	*0.7*	*0.748*	*0.755*	*0.751*	*0.72*	*0.713*	*0.716*	***0.735***
SUM	RTE3	0.6	0.612	0.732	0.667	0.545	0.41	0.468	0.59
	–	0.55	0.612	0.804	0.695	0.585	0.352	0.439	**0.605**
Overall	RTE3	–	0.615	0.67	0.641	0.618	0.559	0.587	**0.616**
	–	–	0.609	0.761	0.677	0.659	0.487	0.56	**0.628**

Comparison of our obtained scores with the results reported in the Third PASCAL Recognizing Textual Entailment Challenge (RTE 3) [41] reveals that out of total 45 system submissions, 19 systems are ahead of us in terms of accuracy.

Since RTE4 dataset does not contain any development set, the optimal thresholds on the RTE2 and RTE3 development sets were used to evaluate the performance of the method on the RTE4 testset. Evaluation results of 2–way TE task on the RTE4 testset are presented in Table 43. In addition to the performances using the optimal thresholds on the RTE2 and RTE3 devsets, oracle performances on the RTE4 testset for each of the 4 tasks are also provided in Table 43. It was observed that the highest and lowest scores were obtained for the IR and IE tasks respectively with the overall accuracy reaching a value of 61.5% for this testset.

Table 43: Evaluation results on the RTE4 testset

Task	**Devset**	**Optimum threshold**	**Positive TE**			**Negative TE**			**Overall Accuracy**
			Pr	**Re**	**F**	**Pr**	**Re**	**F**	
IE	RTE2	0.6	0.542	0.647	0.589	0.562	0.453	0.502	0.55
	RTE3	0.45	0.533	0.853	0.656	0.633	0.253	0.362	0.553
	–	0.5	0.541	0.833	0.656	0.638	0.293	0.402	***0.563***
IR	RTE2	0.6	0.642	0.587	0.613	0.619	0.673	0.645	0.63
	RTE3	0.8	0.806	0.36	0.498	0.588	0.913	0.715	0.637
	–	0.65	0.689	0.56	0.618	0.629	0.747	0.683	***0.653***
QA	RTE2	0.85	0.696	0.32	0.438	0.558	0.86	0.677	0.59
	RTE3	0.7	0.598	0.55	0.573	0.583	0.63	0.606	0.59
	–	0.75	0.645	0.49	0.557	0.589	0.73	0.652	***0.61***
SUM	RTE2	0.65	0.675	0.52	0.587	0.61	0.75	0.673	0.635
	RTE3	0.6	0.656	0.59	0.621	0.627	0.69	0.657	0.64
	–	0.5	0.613	0.76	0.679	0.684	0.52	0.591	***0.64***
Overall	RTE2	–	0.613	0.538	0.573	0.588	0.66	0.622	0.599
	RTE3	–	0.605	0.592	0.598	0.601	0.614	0.607	0.603
	–	–	0.604	0.668	0.634	0.629	0.562	0.594	***0.615***

Comparison of the proposed system performance with the systems submitted in the Fourth PASCAL Recognizing Textual Entailment Challenge[10] (RTE 4) [42] in measuring 2–way entailment task reveals that among 45 submissions only 7 systems exhibited higher accuracy than our method.

The T–H pairs in the RTE4 testset are also classified for the 3–way RTE task with 3 class levels: ENTAILMENT (E), CONTRADICTION (C), and UNKNOWN (U). Table 44 shows the performance of the method in labelling the T–H pairs in 3-way RTE task. In order to label the T–H pairs for 3–way classification, we set 2 threshold values TH_U and TH_L. The entailment score generated for any T–H pair is compared against TH_U first. If the entailment score exceeds TH_U, the given T–H pair is marked as E. Otherwise, the score is then compared against TH_L. If the score falls below TH_L, the given pair is marked as a case of C. A score lying in between TH_L and TH_U marks the pair as U. TH_U was already learned from Table 43. The value of TH_L is varied in the range of -0.9 and -0.1. The optimal performance obtained for the 3–way entailment classification task on the RTE4 testset is summarized in Table 44.

[10] http://tac.nist.gov/publications/2008/index.html

Table 44: Evaluation results on RTE4 testset for the 3–way entailment task (Th: Threshold, T: Total, C: Correctly labelled, Acc: Accuracy)

Task	Th_U	Th_L	Entailment			Contradiction			Unknown			Overall		
			T	C	Acc	T	C	Acc	T	C	Acc	T	C	Acc
IR	0.65	-0.2	150	84	0.56	45	10	0.22	105	77	0.73	300	171	0.57
QA	0.75	-0.5	100	49	0.49	30	2	0.067	70	49	0.7	200	100	0.5
SUM	0.5	-0.5	100	75	0.75	30	8	0.27	70	36	0.514	200	119	0.595
IE	0.5	-0.3	150	125	0.83	45	3	0.06	105	25	0.238	300	153	0.51
Overall	-	-	500	333	0.67	150	23	0.153	350	187	0.534	1000	543	**0.543**

As can be observed from Table 44, the method exhibits accuracy of 67%, 15.3% and 53.4% for the ENTAILMENT, CONTRADICTION and UNKNOWN classes, respectively, thus attaining an overall accuracy of 54.3% for the 3– way TE problem.

Out of total 66 submissions in RTE4 challenge [42], 33 systems addressed the problem of the 3–way TE task. Out of those 33, 16 systems show better performance than our method.

It is evident from Table 44 that the accuracy for the CONTRADICTION class came out to be very low compared to the ENTAILMENT and UNKNOWN classes. In order to specifically handle the contradictory cases, Rule 4 was designed under the Single Triplet Dependency (STD) category. However, this rule can be successfully applied where two dependency triplets related to each other have an antonym token pair directly associated within them. A thorough analysis of the T–H pairs belonging to the CONTRADICTION class in the RTE4 testset revealed that the way the T–H pairs express the contradiction are very hard to be detected by a dependency parser based method resulting in very low score for this particular class.

6. Analysis

This section comprises of two parts. The first subsection analyses the effectiveness of the various rules designed under lexical triplet matching module (cf. subsection 3.7.1) and semantic similarity matching module (cf. subsection 3.7.2) on the overall performance of the method. The second subsection provides an in-depth error analysis of the proposed method.

6.1 Analysis of several matching rules in overall performance of the method

In order to show the contribution of the matching rules designed under several categories in overall performance of the method, we built 5 different models and tested our method on the RTE3 dataset using each of these 5 models. The 5 models are described in Table 45.

Table 45. Selection of models

Model #	Combination of Components
Model 1	Only identical triplets matching; no rules are applied.
Model 2	All rules of the STD category
Model 3	Model 2 + All rules of the JTD category
Model 4	Model 1+ Rules belonging to semantic similarity matching module
Model 5	The entire system comprising all rules

The T–H pairs of the RTE3 devset are subjected to each of the 5 models separately and the method accuracy is measured varying the threshold between 0.05 and 0.95. Table 46 presents the highest accuracy for each of the 4 tasks as well as the overall performance obtained from applying each of the 5 models. Task specific highest

accuracies obtained from the 5 models are shown in bold in Table 46. The performance analysis using bar graphs are shown in Figure. 30.

Table 46. Task-wise highest accuracies for different models on the RTE3 devset

Task	Model 1	Model 2	Model 3	Model 4	Model 5
IE	46.5	**55.5**	54	53.5	53.5
IR	63	67.5	**69.5**	67	66.5
QA	**73**	72.5	72.5	67.5	71
SUM	**65.5**	65	63.5	65	64.5
Overall	60.38	63.25	**64**	60.5	63.38

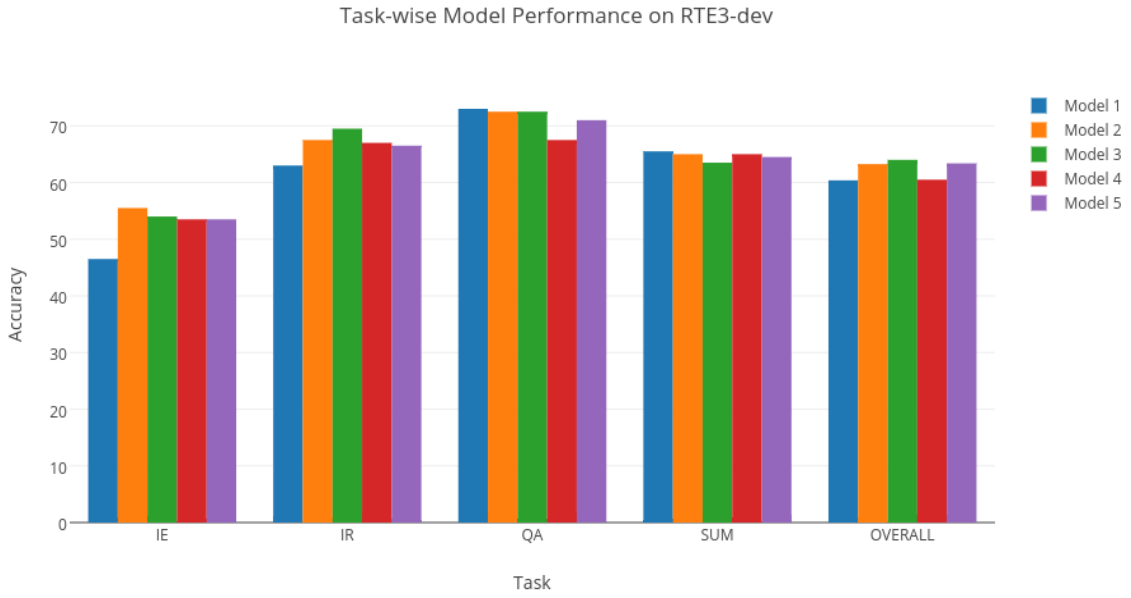

Figure. 30: Task-wise Performance Analysis on the RTE3 development set

As Model 1 considers only identical triplet matching and does not make use of any rules, it serves as our baseline model and hypothetically it should perform the poorest among all the 5 models, which is indeed the case. However, as Figure. 30 shows, this model produces the best results for the QA and SUM tasks. Model 2 significantly improves the overall performance which indicates the effectiveness of the rules in the STD category. Model 2 also produces the best score for the IE task. Model 3 slightly improves over Model 2 and it produces the overall best performance among all the models which denotes the worthiness of the rules under the JTD category. Model 3 also yields in the best scores for the IR task. Model 4 performs marginally better than the baseline model, however, it performs significantly poorer than Model 2 and Model 3. This suggests that the rules belonging to semantic similarity matching module are also useful, but not as much as the STD and JTD rules on the RTE3 devset. The fact that Models 2, 3 and 4 improve over the baseline model (i.e., Model 1) corroborates the contribution of the rules synthesized under the STD, JTD and semantic similarity matching modules. Although according to our philosophy, Model 5, comprising of all the modules, should achieve the highest performance, it was observed that Model 5 performs poorer than Model3. This might be an indication that the combination of STD-JTD and semantic similarity matching rules seems to be hurting the performance of the method on the RTE3 devset.

Each of the 5 models as stated were also applied separately on the RTE3 test set and the task–wise highest performance from the different models and the overall performance are presented in Table 47. Figure. 31 gives a pictorial representation of Table 47 using bar graphs.

Table 47. Task wise highest accuracies using 5 models on RTE3 testset

Task	**Model 1**	**Model 2**	**Model 3**	**Model 4**	**Model 5**
IE	51	54.5	**56**	54	55
IR	60.5	60.5	61.5	**63**	62
QA	65	66	69	65.5	**73.5**
SUM	58	61.5	59	**62.5**	60.5
Overall	58	58.63	59.25	58.88	**62.8**

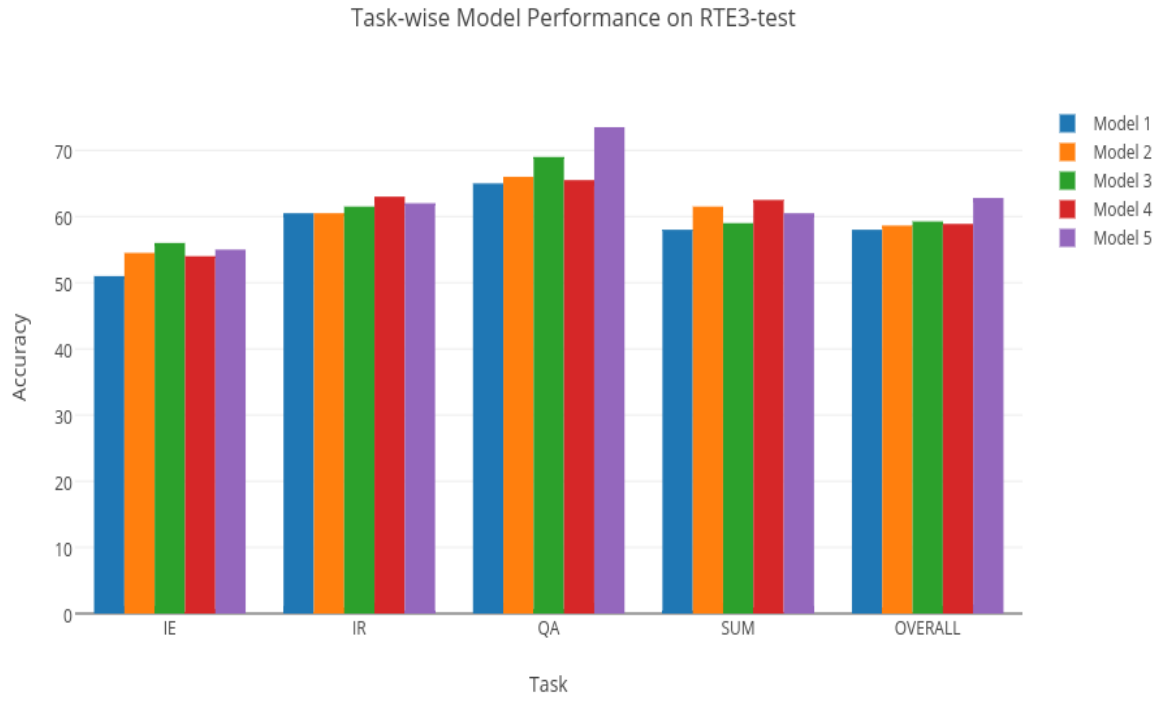

Figure. 31: Task-wise Performance Analysis on RTE3 test set

As can be observed from Figure. 31, Model 2 performs better than Model 1 for almost all the tasks. Model 3 improves the overall method performance over Model 2. Model 4 improves over the baseline model. In contrast to the results on the RTE3 devset, Model 4 provides better results than Model 2 on the RTE3 test set. Similarly, unlike in case of the RTE3 devset, Model 5 produces the overall best performance.

6.2 Error Analysis

We manually analyzed a subset of the T–H pairs in the development sets and test sets for which the method resulted in false positives and false negatives, which in turn diminish the performance of the method. We observed three sources of errors which are discussed below.

- Errors in RTE datasets
- Errors of the embedded tools
- Error of the proposed method

6.2.1 Errors in RTE datasets There are many typographical errors scattered among several data pairs in the RTE datasets. An example data pair from the RTE2 development set containing such error is shown below with the erroneous word highlighted in bold font. The adverse effects of such errors on calculating the entailment score for a particular T–H pair finally leading to diminishing the ultimate method accuracy is also illustrated.

Example 12:

<pair id="80" entailment="YES" task="QA">
<t>About 33.5 million people live in this massive conurbation. I would guess that 95% of the 5,000 officially foreign-capital firms in Japan are based in Tokyo.</t>
<h>About 33.5 **miilion** people live in Tokyo.</h>
</pair>

Due to the presence of such typing mistakes in the dataset, the dependency parser also generates erroneous outputs. Table 45 shows some of the relevant text and hypothesis dependency triplets for the above T–H pair as produced by the dependency parser.

Table 45: Dependency triplets produced for the T–H pair in Example 12

Text dependency triplets	Hypothesis dependency triplets
quantmod(million-3, About-1)	quantmod(33.5-2, About-1)
number(million-3, 33.5-2)	num(people-4, 33.5-2)
num(people-4, million-3)	nn(people-4, **miilion**-3)

Since in the given T–H pair, the text (T) is free from typos, the corresponding text dependency triplets as shown in the first column of Table 45 are all correct. However, the typographical error existing in the hypothesis permeates in dependency parsing resulting in incorrect triplets as shown in the second column of the table, which do not match with any of the text triplets leading to generating a low entailment score (0.45) which falls below the threshold value. Therefore, the method incorrectly labels the above T–H pair as a case of NO entailment. It is evident that this type of error occurs entirely due to the problems in the dataset itself and not due to the proposed method.

Since the RTE datasets serve as standard datasets, for a fair comparison we tested the method on the dataset without making any corrections to these erroneous words or using any text normalizer. Therefore, the presence of such typographical errors decreases the accuracy of our method to a certain extent despite the fact that this is not a drawback of our method per se. A few examples having such typing errors from the various RTE datasets are provided in Table 46 with the erroneous words highlighted in bold fonts.

Table 46: Examples of erroneous T–H pairs in the RTE datasets

RTE dataset#	T–H pair
RTE1 dev1	<pair id="56" value="TRUE" task="IR"> <t>Euro-Scandinavian media cheer Denmark **v** Sweden draw.</t> <h>Denmark and Sweden tie.</h> </pair>
RTE2 dev	<pair id="436" entailment="YES" task="QA"> <t>Edward VIII shocked the world in 1936 when he gave up his throne to marry an American divorcee, Wallis Simpson.</t> <h>King Edward VIII **abdictated** in 1936.</h> </pair>
RTE2 dev	<pair id="740" entailment="YES" task="QA">

	<t>Edwin Hubble is **recongized** as having been one of the foremost astronomers of the modern era.</t> <h>Edwin Hubble was an astronomer. </h> </pair>
RTE2 dev	<pair id="664" entailment="YES" task="SUM"> <t>Radical Jordanian cleric, Abu Qatada, and nine other foreign nationals said to pose a threat to the UK's security, have been detained, pending deportation.</t> <h>10 foreign nationals were a threat to **Britian**'s national security. </h> </pair>
RTE2 dev	<pair id="15" entailment="YES" task="IR"> <t>A juvenile hacker who crippled an airport tower for six hours, damaged **atown's** phone system, and broke into pharmacy records has been charged in **afirst**-ever federal prosecution, the U.S. Attorney's office **announcedtoday**. </t> <h>Non-authorized personnel illegally entered into computer networks.</h> </pair>
RTE2 dev	<pair id="630" entailment="YES" task="IR"> <t>These early men learned to make fire. They **traveled** over land bridges from Africa, and began to populate the world, about 1 million years ago. </t> <h>Humans existed 10,000 years ago.</h> </pair>
RTE3 dev	<pair id="728" entailment="YES" task="SUM" length="short" > <t>The Arak plant, along with the discovery of a secret Iranian enrichment program in 2003, Tehran's refusal to cease uranium enrichment and findings by IAEA inspectors have increased suspicions about **Iran s** program**.**</t> <h>Iran's program is under suspicion because of the findings by IAEA inspectors.</h> </pair>

6.2.2 Errors of the embedded tools The proposed method makes use of the following NLP tools.

- Stanford dependency parser
- Stanford POS tagger
- Stanford stemmer
- WordNet 2.1
- RiTa
- WordNet::Similarity package

Some of these tools produce some incorrect outputs for some cases which ultimately degrades the accuracy of the proposed method. A few example T–H pairs from the RTE datasets are provided in Table 47 for which the Stanford dependency parser produces erroneous output.

Table 47: Examples of T–H pairs for which dependency parser makes mistakes

Pair Id#	Dataset	T–H pairs
1	RTE1-dev2	T: Blair has sympathy for anyone who has lost their lives in Iraq. H: Blair is sympathetic to anyone who has lost their lives in Iraq.
2	RTE2-dev	T: American illusionist, James Randi, offered $1m to anyone able to prove, under observed conditions in a laboratory, that homeopathic remedies can really cure people. H: Illusionist James Randi offered a million dollars to anyone able to prove that homeopathy cures.
3	RTE3-dev	T: India and Pakistan have agreed to release hundreds of fishermen and other civilians in each other's jails, a goodwill measure that comes as part of a peace process between the two countries. H: India and Pakistan have decided to free hundreds of civilian prisoners in each others jails.

Table 48: Outputs of the Stanford dependency parser for the examples in Table 47

Pair id #	Text Dependency triplets	POS-tagged text tokens	Hypothesis Dependency triplets	POS-tagged hypothesis tokens
1	T1: nsubj(has-2, Blair-1) T2: dobj(has-2, sympathy-3) **T3: nsubj(lost-8, sympathy-3)** **T4: rcmod(sympathy-3, lost-8)**	lost-VBN sympathy-NN	H1:nsubj(sympathetic-3, Blair-1) H2: nsubj(lost-8, anyone-5) H3: rcmod(anyone-5, lost-8)	lost-VBN sympathetic-JJ
2	T1:amod(remedies-25, homeopathic-24) T2: nsubj(cure-28, remedies-25)	remedies-NNS cure- VB	**H1:amod(cures-15,homeopathy-14)** **H2:dobj(prove-12, cures-15)**	**cures-NNS** prove-VB
3	T1: xcomp(agreed-5, release-7) T2: dobj(release-7, hundreds-8)	agreed-VBN release-VB	**H1:amod(hundreds-8, free-7)** **H2:prep_to(decided-5, hundreds-8)**	**free-JJ** decided-VBN

Table 48 presents a few of the dependency triplets produced by the Stanford parser for the T–H pairs presented in Table 47 with the erroneous dependency triplets shown in bold fonts. The reason behind generating these wrong dependency triplets by the Stanford parser is errors made by the Stanford POS tagger which propagate in dependency parsing. The POS-tagged outputs of the Stanford POS tagger are also shown in Table 49 with the wrong POS tags being highlighted in bold fonts.

Due to the incorrect dependency triplets generated by the parser, the correct hypothesis triplets H2 and H3 of Pair id 1 do not match with any of the text dependency triplets. For the same reason the incorrect hypothesis triplets H1 and H2 of pair id 2 do not satisfy any matching criterion for match with any of the correct text dependency triplets. The same holds true for pair id 3. Therefore, the errors of the embedded tools ultimately propagate to the final step where the entailment decision is taken.

WordNet 2.1, the lexical database which has been used as the underlying lexical resource in our method, does not provide sufficient coverage for antonym relations. A few example T–H pairs from the RTE devsets are provided in Table 49 to illustrate the effect of this on our method.

Table 49: Example T–H pairs from RTE datasets

RTE dataset#	T–H pair
RTE1 dev1	<pair id="148" value="FALSE" task="RC"> <t>The Philippine Stock Exchange Composite Index **rose** 0.1 percent to 1573.65.</t> <h>The Philippine Stock Exchange Composite Index **dropped**.</h> </pair>
RTE1 dev2	<pair id="1950" value="FALSE" task="IE"> <t>Crude oil **dips** below $43 on news that Russia's justice ministry will not force Yukos to halt sales.</t> <h>Crude oil **rises.**</h> </pair>

The tokens which are highlighted in bold fonts in the T–H pairs in Table 49 are antonyms of each other. However, due to inadequate coverage of antonyms in WordNet 2.1, they cannot be lexically aligned with each other. The WordNet::Similarity package is invoked which returns Wu-Palmer semantic scores of 0.35 and 0.33 for the word pairs *rise–drop* and *dip–rise* respectively. If those word pairs can be lexically aligned to each other by the antonym relation, then rule 4 designed under the STD category (cf. subsection 3.7.1) can be applied to assign the scores for these cases and the final entailment scores for the above two T–H pairs would be much less than the threshold which would correctly label the pairs as cases of NO entailment.

6.2.3 Error of the method

The errors made the proposed method can be classified into two types: false positives, where the method incorrectly labels a T–H pair as a case of YES entailment, and false negatives, where a T–H pair is incorrectly

labelled as a case of NO entailment. The reasons behind these two types of errors are explained with suitable examples.

- **False positive:** There are many T–H pairs in the RTE datasets where each and every dependency triplet of the hypothesis matches with some text triplets satisfying any of the matching criterions (cf. section 3.7), resulting in a very high entailment score. Such high scores above the threshold value incorrectly label those T–H pairs as cases of YES entailments. An example T–H pair from the RTE1-dev1 set is provided below and the corresponding dependency parser outputs are presented in Table 50.

  Example 13:

  <pair id="1902" value="FALSE" task="IE">
  <t>Sonia Gandhi can be defeated in the next elections in India by BJP.</t>
  <h>Sonia Gandhi is defeated by BJP.</h>
  </pair>

Table 50: Sample dependency triplets for the T–H pair in Example 13

Text dependency triplets	Hypothesis dependency triplets
T1: nn(Gandhi-2, Sonia-1) **T2:** nsubjpass(defeated-5, Gandhi-2) **T3:** aux(defeated-5, can-3) **T4:** auxpass(defeated-5, be-4) **T5:** root(ROOT-0, defeated-5) **T6:** det(elections-9, the-7) **T7:** amod(elections-9, next-8) **T8:** prep_in(defeated-5, elections-9) **T9:** prep_in(elections-9, India-11) **T10**:agent(defeated-5, BJP-13)	**H1:** nn(Gandhi-2, Sonia-1) **H2:** nsubjpass(defeated-4, Gandhi-2) **H3:** auxpass(defeated-4, is-3) **H4:** root(ROOT-0, defeated-4) **H5:** agent(defeated-4, BJP-6)

Table 51: Simulation of matching rules

Hypothesis triplet#	Lexically matched with Text triplet#	According to matching rule#	Score assigned to *P* vector	Score assigned on node
H1	T1	1	1.0	Sonia
H2	T2	1	1.0	Gandhi
H3	T4	1	1.0	Is
H4	NA (ROOT node)	–	0	Defeat
H5	T10	1	1.0	BJP

Figure. 30 depicts the hypothesis graph where each node is assigned a value of 1.0 in its *P* score component according to the matching as shown in Table 51. The score components of each node are shown as an ordered set of quadruple where the 4 values in sequence represent {*P*, *A*, *C*, *T*} respectively (cf. Section 3.8).

It is evident from the diagram that this T–H pair is marked as a case of YES entailment since the final entailment score accumulated at the ROOT node is 1.0. A few more examples of such T–H pairs for which the method generates false positives are given in Table 52. The proposed method cannot correctly detect these pairs as cases of NO entailments as each and every hypothesis dependency triplet finds its corresponding matching text triplet thereby assigning a *P* score of 1.0 to all the nodes of the hypothesis graph.

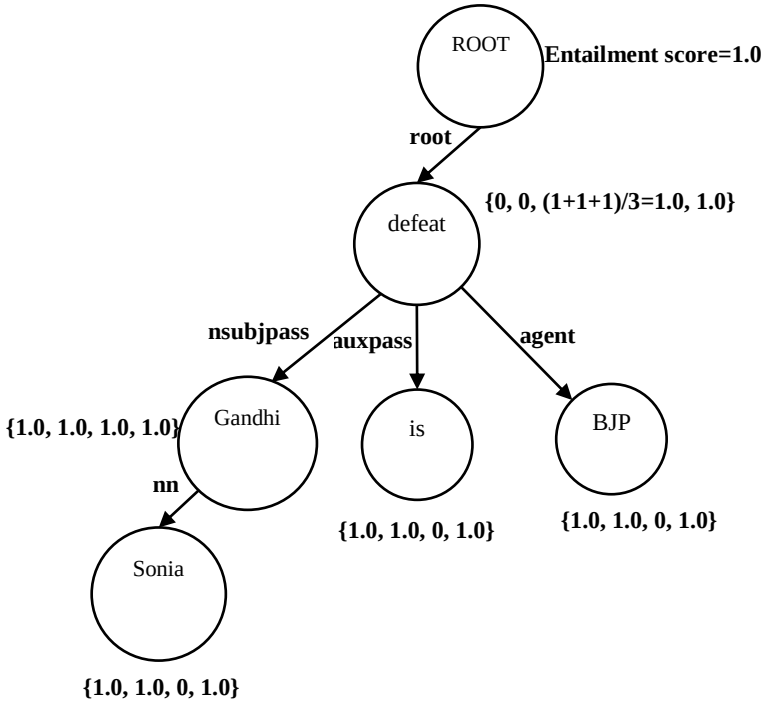

Figure. 30. Hypothesis graph scoring for the T–H pair in Example 13

Table 52: Examples T–H pairs in RTE datasets resulting in false positives

Pair Id#	RTE dataset #	T–H pair
1	RTE1-dev2	<pair id="2065" value="FALSE" task="QA"> <t>The slender tower is the **second tallest** building in Japan.</t> <h>The slender tower is the **tallest** building in Japan.</h> </pair>
2	RTE1-dev2	<pair id="2064" value="FALSE" task="QA"> <t>The Osaka World Trade Center is the tallest building in **Western Japan**.</t> <h>The Osaka World Trade Center is the tallest building in **Japan**.</h> </pair>
3	RTE2 dev	<pair id="54" entailment="NO" task="IR"> <t>By the time a case of rabies is confirmed, the disease **may have** taken hold in the area. </t> <h>A case of rabies was confirmed.</h> </pair>
4	RTE1 dev2	<pair id="2080" value="FALSE" task="QA"> <t>VCU School of the Arts In **Qatar is located in Doha**, the capital city of Qatar.</t> <h>**Qatar is located in Doha**.</h> </pair>
5	RTE2 dev	<pair id="406" entailment="NO" task="QA"> <t>In 2000 there were 10,578 divorces in Bulgaria, which represents 301 divorces per 1000 marriages or a 1.3% divorce rate per 1000 inhabitants. </t> <h>In 2000 there were 301 divorces in Bulgaria.</h> </pair>
6	RTE2 dev	<pair id="413" entailment="NO" task="QA"> <t>A male rabbit is called a buck and a female rabbit is called a doe, just like deer.</t> <h>A female rabbit is called a buck. </h> </pair>

For the pair id 1 in Table 52, the superlative token *tallest* is associated with a modifier *second* in the text which is not present in the hypothesis. Similarly for the pair id 2, the argument *Japan* of the superlative word *tallest* is associated with the word *western* as a modifier in the text which is not present in the hypothesis. The text in the pair id 3 imparts a probabilistic situation by the highlighted phrase "*may have*" which is not captured by the method. In pair id 4, the entire hypothesis is contained within the text as highlighted. For each of the cases in pair id 5 and 6, although the text conveys different meaning from the hypothesis, all the hypothesis dependency triplets generated can be found in the text. For each of the T–H pairs listed in Table 52, all the hypothesis triplets match with some text triplets, finally generating an entailment score of 1. Therefore, the method incorrectly labels each of the above pairs as cases of YES entailments. N–gram matching or finding the longest common subsequence from a given T–H pair might be able to handle this problem to a certain extent.

The JTD category which was designed under the lexical triplet matching module (cf. subsection 3.7.1) also produces some false positive outputs. To illustrate this issue, an example T–H pair from the RTE dataset is presented in Table 53.

Table 53: Examples of false positive T–H pairs under JTD category

R^H	R^{T_1}	R^{T_2} (JDC)	T–H pairs	Dependency triples
nsubjpass	nsubjpass	nn	<pair id="2084" value="FALSE" task="QA"> <t>Microsoft Israel was founded in 1989 and became one of the first Microsoft branches outside the USA.</t> <h>Microsoft was established in 1989.</h> </pair>	nsubjpass(founded-4, Israel-2) nn(Israel-2, Microsoft-1) nsubjpass(established-3, Microsoft-1)

Although the relation **nn** has been identified as a Joint Dependency Compatible (JDC) relation under the JTD category 1, for this particular case (cf. Table 53), the method incorrectly labels it as a case of YES entailment.

- **False negatives** There are also cases of T–H pairs for which the method cannot capture the similarity in dependencies properly and assigns unduly low scores which fall below the threshold value and therefore marks such T–H pairs as cases of NO entailments. One such example from the RTE1 dev1 set is given below.

  Example 14:

  <pair id="58" value="TRUE" task="IR">
  <t>Iraqi militants said Sunday they would behead Kim Sun-Il, a 33-year-old translator, within 24 hours unless plans to dispatch thousands of South Korean troops to Iraq were abandoned.</t>
  <h>translator kidnapped in Iraq</h>
  </pair>

  Although a human reading of the text of the above T–H pair passively infers that the *translator has been kidnapped* but this fact is not clearly conveyed in the text to be easily understood by a dependency parsing based method. A close look at the corresponding T–H dependency triplets provided in Table 54 indicates that none of the hypothesis dependency triplets H1, H2 and H3 matches with any of those in the text, finally generating an entailment score of 0.

  Table 54: Dependency triplets produced by the Stanford parser for the T–H pair in Example 14

Text dependency triplets	Hypothesis dependency triplets
amod(militants-2, Iraqi-1) nsubj(said-3, militants-2) root(ROOT-0, said-3) tmod(said-3, Sunday-4) nsubj(behead-7, they-5) aux(behead-7, would-6) dep(said-3, behead-7)	**H1:**nsubj(kidnapped-2, translator-1) **H2:**root(ROOT-0, kidnapped-2) **H3**:prep_in(kidnapped-2, Iraq-4)

nn(Sun-Il-9, Kim-8) dobj(behead-7, Sun-Il-9) det(translator-13, a-11) amod(translator-13, 33-year-old-12) appos(Sun-Il-9, translator-13) num(hours-17, 24-16) prep_within(Sun-Il-9, hours-17) mark(abandoned-30, unless-18) nsubjpass(abandoned-30, plans-19) aux(dispatch-21, to-20) infmod(plans-19, dispatch-21) dobj(dispatch-21, thousands-22) amod(Korean-25, South-24) amod(troops-26, Korean-25) prep_of(thousands-22, troops-26) prep_to(dispatch-21, Iraq-28) auxpass(abandoned-30, were-29) advcl(behead-7, abandoned-30)	

There are many cases where the proposed method generates low entailment scores due to the reason stated above and thus fails to correctly label those T–H pairs as cases of YES entailments. A few such examples are provided in Table 55.

Table 55: T–H pairs in RTE datasets that result in false negatives

RTE dataset #	**T–H pair**
RTE1 dev1	<pair id="85" value="TRUE" task="IR"> <t>The country's largest private employer, Wal-Mart Stores Inc., is being sued by a number of its female employees who claim they were kept out of jobs in management because they are women.</t> <h>Wal-Mart sued for sexual discrimination</h> </pair>
RTE1 dev2	<pair id="1658" value="TRUE" task="IE"> <t>DAYTON, Ohio. A cargo plane bound for Montreal with a small quantity of hazardous chemicals crashed and exploded shortly after takeoff.</t> <h>Dayton is located in Ohio.</h> </pair>
RTE1 dev1	<pair id="64" value="TRUE" task="IR"> <t>The wait time for a green card has risen from 21 months to 33 months in those same regions.</t> <h>It takes longer to get green card.</h> </pair>
RTE2 dev	<pair id="94" entailment="YES" task="IR"> <t>If legalization reduced current narcotics enforcement costs by one-third to one-fourth, it might save $6 - $9 billion per year. </t> <h>Drug legalization has benefits.</h> </pair>

For each of the pairs listed in the table above, the text conveys the same meaning as that of the hypothesis, but in a totally different way which is very hard to be detected by a machine. Therefore, they result in false negatives.

In the lexical alignment phase (cf. section 3.5), we considered only one WordNet lexical relation to align a token of the hypothesis with any text token. However, there are some cases where a combination of 2 or more lexical relations can align a token pair. An example T–H pair from the RTE dataset is presented in Table 56 to demonstrate this phenomenon.

Table 56: Example T–H pair depicting the need of combining lexical relations for alignment of tokens

RTE dataset#	T–H pair	Dependency triplets
RTE1 dev1	<pair id="570" value="TRUE" task="QA"> <t>"I guess you have to expect this in a growing community"; said Mardelle Kean, who lives across the street from John Joseph Famalaro, charged in the **death** of Denise A. Huber, who was 23 when she disappeared in 1991.</t> <h>John J. Famalaro is accused of having **killed** Denise A. Huber.</h> </pair>	prep_of(**death**-32, Huber-36) dobj(**killed**-8, Huber-11)

The words '*kill*' and '*death*' as highlighted in the above table can be aligned to each other in the following way as depicted in the Figure. 31. Although the relations 'prep_of' and 'dobj' of the generated dependency triplets are equivalent to each other by the matching rule 2 under STD category (cf. subsection 3.7.1), still the triplets do not match with each other as the token pair *kill–death* cannot be lexically aligned in a single step.

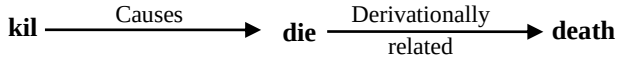

Figure. 31. Combining lexical relations for alignment of tokens

There are many T–H pairs in the RTE datasets which can be properly labelled as cases of YES entailments only with the help of some logic driven methods like First Order Predicate Logic (FOPL) or employing some inference rule based techniques such as DIRT (Discovery of Inference Rules from Text) [34] etc. It is very difficult for a dependency parser based method alone to solve these cases correctly. A few examples are presented in Table 57 which can only be properly tackled by methods based on logical inferences.

Table 57: Example of T–H pairs that can be handled by logic driven techniques

RTE dataset#	T–H pair
RTE2 dev	<pair id="231" entailment="YES" task="IE"> <t>Speaking of Jean Charles de Menezes, who was chased by armed officers into the station and shot five times at close range**, his cousin,** Alex Pereira, hinted today that his family would sue Scotland Yard over the killing.</t> <h>Jean Charles de Menezes **is related to** Alex Pereira.</h> </pair>
RTE2 dev	<pair id="290" entailment="YES" task="IE"> <t>Kaspars Ruklis, **press official** at the United States Embassy, told the Baltic News Service that Mrs. Bush chose to visit Latvia's Occupation.</t> <h>Kaspars Ruklis **works for** the United States Embassy.</h> </pair>
RTE2 dev	<pair id="332" entailment="YES" task="IE"> <t>Kevin Whitaker, **who heads the** Cuban affairs office at the Department of State, spoke with Lazo on two occasions about the effort to give his sons visas.</t> <h>Kevin Whitaker **is a manager of** the Department of State.</h> </pair>

In the first example in Table 57, it is hard to detect for the method that if *A is the cousin of B* then *A is related to B*. Similarly in the 2nd example, *if A is the press official at B,* only a logic driven tool can determine that *A works for B*. In the 3rd case, the fact that *A is a manager of the company B* cannot be inferred from the statement *A heads the company B* by a dependency parser based method.

The proposed method also faces problems in aligning multi-word expressions (MWE). Some examples are provided in Table 58 to illustrate the issue. The MWEs in the given T–H pairs are highlighted in bold fonts. In

spite of the existence of these MWEs in the WordNet, the lexical resource which has been used in the present work for the purpose of lexical alignment, problems still arise because of the presence of ambiguous entities.

Table 58: Examples of T–H pair containing multi-word expressions

Pair Id#	RTE dataset#	T–H pair
1	RTE1 dev1	<pair id="131" value="TRUE" task="IR"> <t>Wasp and bee stings can be **life threatening** if you are allergic to the venom.</t> <h>bee stings can be **fatal.**</h> </pair>
2	RTE2 dev	<pair id="585" entailment="YES" task="IR"> <t>Since the fear of death is virtually a universal phenomenon, the **death penalty** is an unparalleled deterrent for people considering a crime.</t> <h>**Capital punishment** is a deterrent to crime.</h> </pair>
3	RTE1- dev1	<pair id="570" value="TRUE" task="QA"> <t>“I guess you have to expect this in a growing community”; said Mardelle Kean, who lives across the street from **John Joseph Famalaro**, charged in the death of Denise A. Huber, who was 23 when she disappeared in 1991.</t> <h>**John J. Famalaro** is accused of having killed Denise A. Huber.</h> </pair>

For the pair id 1 in Table 58, the two tokens *life* and *threatening* both are individual entities themselves. Since the dependency parser does not provide any information about MWEs, it is hard to detect *life threatening* as an MWE. If *life threatening* is identified as an MWE, then it becomes trivial to compute the semantic similarity between *life threatening* and *fatal* and align them accordingly. For the paid id 2, both *capital punishment* and *death penalty* are MWEs and are present in the WordNet as synonyms; still they cannot be lexically aligned since the dependency parsing based method cannot detect them as MWEs. Finally for the pair id 3, both the MWEs refer to the same named entity. However, since the name is abbreviated in the hypothesis, it cannot be aligned with its full form in the text. In addition to MWE detection, this T–H pair requires entity disambiguation module to correctly classify this pair.

For this same reason as stated above, problem arises in matching acronyms to their corresponding expanded forms. Although WordNet contains some acronyms, however, there are many acronyms scattered in the various RTE datasets which are not available in the WordNet. Irrespective of whether the acronyms are present in the WordNet or not, the dependency relation based method cannot align these acronyms to their corresponding full forms. A few example T–H pairs from the RTE datasets containing such acronyms are shown in Table 59. The abbreviated forms and their corresponding expanded forms are highlighted in bold font.

Table 59: Examples of T–H pairs containing acronyms

RTE dataset #	T–H pair	Existence in WordNet
RTE1 dev1	<pair id="569" value="TRUE" task="QA"> <t>Government forces killed the head of the **Armed Islamic Group,** or **GIA**, which has claimed responsibility for killing 61 foreigners in the last year.</t> <h>The abbreviation **GIA** stands for Armed Islamic Group.</h> </pair>	Yes
RTE1 dev1	<pair id="357" value="TRUE" task="MT"> <t>The funeral procession headed to the headquarters of the **United Nations Development Programme** in Baghdad, where funeral goers held banners denouncing the Americans as “Enemies of God”.</t> <h>The funeral procession had been directed to the **UNDP** headquarters in Baghdad, where signs denouncing Americans as “enemies of God” were seen.</h> </pair>	No

RTE1 dev1	<pair id="520" value="TRUE" task="PP"> <t>Only 18 states, including Massachusetts, have specific provisions that allow women who quit their jobs due to domestic violence to qualify for **UI**.</t> <h>18 states have provisions that permit women who quit their jobs because of domestic violence to collect **unemployment insurance**.</h> </pair>	No
RTE3 dev	<pair id="754" entailment="YES" task="SUM" length="short" > <t>A team from the U.S. **Centers for Disease Control** boarded the ship when it docked in St. Maarten to oversee the cleaning operation and try to determine what caused the outbreak, Carnival said.</t> <h>The causes of the outbreak were searched for by a team from the U.S. **CDC**.</h> </pair>	Yes
RTE1 dev2	<pair id="2170" value="FALSE" task="CD"> <t>**Hacking** reported his wife missing on July 19, a Monday.</t> <h>**Mark Hacking** was booked into the Salt Lake County Jail on Monday.</h> </pair>	Named Entity

7. Conclusions and Future work We presented a dependency graph based textual entailment recognition method which combines lexical, syntactic and semantic features. After being subjected to a number of preprocessing operations, each of the text fragments of the T–H pair is individually parsed by a dependency parser to generate a dependency graph. The stemmed hypothesis tokens are then aligned with one or more stemmed text tokens based on WordNet relations. Each of the generated hypothesis dependency triplet is compared against all the text triplets to find a corresponding matching pair. This matching operation is carried out on the basis of a set of matching rules which differ from each other on the basis of syntactic divergence between the lexically aligned tokens. However, if for any matched pair of dependency triplets, either of the governor and dependent text-hypothesis token pair is not aligned with each with other, semantic similarity module is invoked to find the most similar text token for that hypothesis token. After this matching module assigns a matching score (lying in the range of -1 to 1) to each node of the hypothesis dependency graph, the graph is traversed in level order fashion by accumulating the scores of the nodes starting from the bottommost level, gradually propagating the scores to the next higher levels until it reaches the topmost level where the final entailment score between the T–H pair is obtained at the root node. The score above a predetermined threshold labels the T–H pair as a case of YES entailment, otherwise it is marked as NO entailment. The experimental results show that our proposed method, being a combination of lexico-syntactic-semantic approach, is quite efficient in correctly taking the entailment decisions for a large portion of the RTE datasets. The method, although essentially a rule-based one, surpasses the accuracy of many methods which use other techniques such as semantic role labelling, logic driven tools, UNL dependency parser and machine learning based approaches found in the literature.

The efficiency of the method can be further improved by overcoming the present limitations of aligning the multi word expressions, matching abbreviated named entities, handling the superlative degrees, combining multiple WordNet relations for lexical alignment of the tokens, mapping the acronyms to their corresponding expanded forms, etc. We would also integrate coreference resolution in our method for replacing the occurrence of the anaphors with their corresponding referring nouns or named entities. Due to memory constraints of the machine, we were not able to employ this tool for this present work. Moreover we have limited the maximum number of text dependency triplets jointly inferring one hypothesis triplet to two only which can be extended later to three or even more. In future, we plan to augment the method with MWE identification and entity disambiguation modules in order to efficiently handle the problem of aligning multiword expressions. We would also like to augment the method with logic driven tools to improve performance.

Moreover, the rigorous effort which has been put to synthesize the matching rules will not be limited to this particular work only; rather experiments are being carried out for fruitful extension of this work to supervised

machine learning framework where these developed matching rules are being used as sophisticated features for the classification task.

References.

Bar Haim, Roy, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. 2006.

Basak, Rohini, Sudip Kumar Naskar, Partha Pakray, and Alexander Gelbukh. Recognizing textual entailment by soft dependency tree matching. Computación y Sistemas, 19(4):685–700, 2015.

Blake, Catherine. The role of sentence structure in recognizing textual entailment. In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pages 101–106. Association for Computational Linguistics, 2007.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising textual entailment, pages 177–190. Springer, 2006.

Delmonte, Rodolfo, Sara Tonelli, Marco Aldo Piccolino Boniforti, and Antonella Bristot. Venses–a linguistically-based system for semantic evaluation. In Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment, pages 344–371. Springer, 2006.

Dinu, Georgiana and RuiWang. Inference rules and their application to recognizing textual entailment. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pages 211–219. Association for Computational Linguistics, 2009.

Giampiccolo, Danilo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 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, 2007.

Giampiccolo, Danilo, Hoa Trang Dang, Bernardo Magnini, Ido Dagan, Elena Cabrio, and Bill Dolan. The Fourth PASCAL Recognizing Textual Entailment Challenge. In TAC. Citeseer, 2008.

Haghighi, Aria D, Andrew Y Ng, and Christopher D Manning. Robust textual inference via graph matching. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pages 387–394. Association for Computational Linguistics, 2005.

Herrera, Jesús, Anselmo Penas, and Felisa Verdejo. Textual entailment recognition based on dependency analysis and wordnet. In Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment, pages 231–239. Springer, 2006.

Hickl, Andrew, JohnWilliams, Jeremy Bensley, Kirk Roberts, Bryan Rink, and Ying Shi. Recognizing textual entailment with LCC's GROUNDHOG system. In Proceedings of the Second PASCAL Challenges Workshop, volume 18, 2006.

Kouylekov, Milen and Bernardo Magnini. Recognizing textual entailment with tree edit distance algorithms. In Proceedings of the First Challenge Workshop Recognising Textual Entailment, pages 17–20, 2005.

Lin, Dekang and Patrick Pantel. DIRT@ SBT@ discovery of inference rules from text. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 323–328. ACM, 2001.

Marsi, EC, EJ Krahmer, WE Bosma, and Mariët Theune. Normalized alignment of dependency trees for detecting textual entailment. 2006.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119, 2013.

Pakray, Partha, Sivaji Bandyopadhyay, and Alexander Gelbukh. Dependency parser based textual entailment system. In Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on, volume 1, pages 393–397. IEEE, 2010a.

Pakray, Partha, Alexander Gelbukh, and Sivaji Bandyopadhyay. A syntactic textual entailment system based on dependency parser. In International Conference on Intelligent Text Processing and Computational Linguistics, pages 269–278. Springer, 2010b.

Pakray, Partha, Utsab Barman, Sivaji Bandyopadhyay, and Alexander Gelbukh. A statistics based semantic textual entailment system. Advances in Artificial Intelligence, pages 267–276, 2011a.

Pakray, Partha, Soujanya Poria, Sivaji Bandyopadhyay, and Alexander Gelbukh. Semantic textual entailment recognition using UNL. Polibits, (43):23–27, 2011b.

Pérez, Diana and Enrique Alfonseca. Application of the Bleu algorithm for recognising textual entailments. In Proceedings of the First Challenge Workshop Recognising Textual Entailment, pages 9–12. Citeseer, 2005.

Rios, Miguel and Alexander Gelbukh. Recognizing textual entailment with a semantic edit distance metric. In Artificial Intelligence (MICAI), 2012 11th Mexican International Conference on, pages 15–20. IEEE, 2012.

Snow, Rion, Lucy Vanderwende, and Arul Menezes. Effectively using syntax for recognizing false entailment. In Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, pages 33–40. Association for Computational Linguistics, 2006.

Tatu, Marta, Brandon Iles, John Slavick, Adrian Novischi, and Dan Moldovan. Cogex at the second recognizing textual entailment challenge. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, pages 104–109, 2006.