JU_CSE_TAC: Textual Entailment Recognition System at TAC RTE-6

Partha Pakray, Santanu Pal, Soujanya Poria,
Sivaji Bandyopadhyay

Computer Science and Engineering Department,
Jadavpur University
Kolkata, India.
{parthapakray, santanu.pal.ju, soujanya.poria}@gmail.com
sivaji_cse_ju@yahoo.com

Alexander Gelbukh

Center for Computing Research,
National Polytechnic Institute,
Mexico City, Mexico.
gelbukh@gelbukh.com

Abstract

The note describes the Recognizing Textual Entailment (RTE) system developed at the Computer Science and Engineering Department, Jadavpur University, India. In this competition, we participated and submitted the results in the RTE-6 Main Task (3 runs), Novelty Task (3 runs) and RTE-6 KBP task (3 runs for generic task and 3 runs for tailored task). For the Main and the Novelty Tasks, the corpus was a collection of news wire documents from various sources and arranged into particular topics, a hypothesis H and a set of "candidate" sentences retrieved by Lucene from that corpus for the hypothesis H. Each sentence in the set of documents associated with a given topic was involved in an entailment relationship with each hypothesis for the topic. RTE systems are required to identify all the sentences that entail H among the candidate sentences.

For the Main and the Novelty Tasks, the system is based on the composition of lexical entailment module, lexical distance module, Chunk module, Named Entity module and syntactic text entailment (TE) module. Our TE system is based on the Support Vector Machine (SVM) that uses twenty five features for lexical similarity, the output tag from a rule based syntactic two-way TE system as a feature and the outputs from a rule based Chunk Module and Named Entity Module as the other features. For the Main task test set, the following micro-average results were obtained for Run 1: F-Score 34.79, Run 2: F-Score 26.78 and Run 3 : F-score 31.19. For the novelty task test set, the following micro-average results were obtained for Run 1: Novelty Evaluation F-Score 81.77 and Justification Evaluation F-Score 34.35, Run 2: Novelty Evaluation F-Score 78.18 and Justification Evaluation 26.87 and Run 3: Novelty Evaluation F-score 78.69 and Justification Evaluation 24.57 were obtained.

The KBP Slot Filling task is focused on the searching a collection of news wire and Web documents and extracting values for a predefined set of attributes (“slots”) for the target entities. The RTE KBP Validation Pilot is based on the assumption that extracted slot filler is correct if and only if the supporting document entails an hypothesis created on the basis of the slot filler. In RTE KBP, we participated for generic task and tailored task. For the RTE-6 KBP test set for Generic Task, micro-average results for Run 1: F-Score 0.1403, Run 2: F-Score 0.172 and Run 3: F-score 0.1531 were obtained. For RTE-6 KBP test set for Tailored Task, micro-average results for Run 1: F-Score 0.3, Run 2: F-Score 0.3307 and Run 3: F-score 0.3288 were obtained.

1. Introduction

The TAC RTE-6 tasks [1] focus on recognizing textual entailment in two application settings: Summarization[1] and Knowledge Base Population[2].

i. Main Task (Summarization scenario): Given a corpus and a set of "candidate" sentences retrieved by Lucene from that corpus, RTE systems are required to identify all the sentences from among the candidate sentences that entail a given Hypothesis. The RTE-6 Main Task is based on the TAC Update Summarization Task.

ii. Novelty Detection subtask: Based on the Main Task, the subtask is focused on Novelty Detection, which means that RTE systems are required to judge whether the information contained in each H is novel with respect to (i.e., not entailed by) the information contained in the corpus.

iii. KBP Validation Pilot (Knowledge Base Population scenario): Based on the TAC Knowledge Base Population (KBP) Slot-Filling task, the new KBP validation pilot task determines whether a given relation (Hypothesis) is supported in an associated document (Text).

Section 2 describes the RTE Data Set and Section 3 describes the RTE system. The various

[1] http://www.nist.gov/tac/2010/RTE/RTE6_Main_NoveltyDetection_Task_Guidelines.pdf
[2] http://www.nist.gov/tac/2010/RTE/RTE6_KBP_Validation_Pilot_Guidelines.pdf

experiments carried out on the development and test data sets are described in Section 4 along with the results. The ablation tests are reported in Section 5. The conclusions are drawn in Section 6.

2. Data Set Description

2.1 RTE-6 Main and Novelty Task Data Set

The RTE-6 Main Task data set is based on the data created for the TAC 2009 Update Summarization task. The TAC 2009 SUM Update data consist of a number of topics, each containing two sets of documents, namely i) Cluster A, made up of the first 10 texts in chronological order (of publication date), and ii) Cluster B, made up of the last 10 texts. The RTE-6 data set is composed of 20 topics, 10 used for the Development Set and 10 for the Test Set.

For each topic, the RTE-6 Main Task data consists of:

a) Up to 30 Hypotheses referring to the topic. H’s are standalone sentences taken from Cluster B documents. When needed, minor syntactic and morpho-syntactic changes have been made with respect to the Cluster B original sentences, from which the H’s are taken, to produce grammatically correct stand alone sentences. Moreover, all the discourse references have been resolved.

b) A set of 10 documents, corresponding to the Cluster A corpus.

c) For each H, a list of up to 100 candidates entailing sentences from the Cluster A corpus and their location in the corpus. The candidate sentences are the 100 top-ranked sentences retrieved by Lucene, using H verbatim as the search query.

DEVELOPMENT SET

The following items were distributed as the Development Set:

The gold standard Development Set and for each topic:

Item A: a list of hypotheses.

Item B: for each hypothesis H, the list of the id numbers of Cluster A candidate sentences to be judged for entailment.

Item C: the set of Cluster A documents for that topic.

Furthermore, H's are not artificially created but are taken from a real text corpus. Cluster B documents, from which the hypotheses are taken, are grouped together in a single file.

The data set distributed for the Novelty Detection task was mostly different from the Main Task data set but was having the same structure.

2.2 RTE-6 KBP Validation Data Set

The RTE-6 KBP Validation data set is based on the data created for the KBP 2009 and 2010 Slot Filling Task. More precisely, the Development Set was created from the KBP slot-filling system output and slot-filler assessments from KBP 2009, whereas the Test Set was created from the corresponding data from KBP 2010[3].

The creation of the RTE-6 Pilot task data set is semi-automatic and takes as starting point (i) the extracted slot-fillers from multiple systems participating in the KBP Slot Filling task and (ii) their assessments.

3. System Description

We submitted 3 runs for Main Task, 3 runs for Novelty Detection sub-task and 6 runs for KBP Validation Pilot task.

3.1 Pre-processing Task

The system accepts pairs of text snippets (text and hypothesis) as the input and gives an entailment value at the output: “YES” if the text entails the hypothesis and “NO” otherwise.

The corpus has some noise as well as some special symbols that create problems during parsing. The list of noise symbols and the special symbols is initially developed manually by looking at a number of documents and then the list is used to automatically replace or remove such symbols from the documents. Table 1 lists the tokens that are replaced by blank as well as by other tokens. All the above pre-processing methods are applied on the development and test set as well.

[3] http://nlp.cs.qc.cuny.edu/kbp/2010/annotation.html

Replace by blank	Replace by Symbol	
	Original Token	Replaced Token
. –	Á	a
();	Č	c
[...]	È	e
()	&amp;	&
...	Š	S
--		

Table 1. Token Replacement List

3.2 Lexical based RTE methods

In this section the various lexical based RTE methods [2] are described in detail.

i) WordNet based Unigram Match: In this method, the various unigrams in the hypothesis for each text-hypothesis pair are checked for their presence in the text. WordNet synsets are identified for each of the unmatched unigrams in the hypothesis. If any synset for the hypothesis unigram matches with any synset of a word in the text then the hypothesis unigram is considered as a WordNet based unigram match.

If n1= common unigram or WordNet Synonyms between text and hypothesis and n2= number of unigram in Hypothesis then Wordnet_Unigram_Match=n1/n2.

If the value of Wordnet_Unigram_Match is 0.75 or more, i.e., 75% or more unigrams in the hypothesis match either directly or through WordNet synonyms, then the text-hypothesis pair is considered as entailment. The text-hypothesis pair is then assigned the value of 1 meaning entailment, otherwise, the pair is assigned the value of 0. The cut-off value for the Wordnet_Unigram_Match is based on experiments carried out on the RTE-6 Main and Novelty Task development set.

ii) Bigram Match: Each bigram in the hypothesis is searched for a match in the corresponding text part. The measure Bigram_Match is calculated as the fraction of the hypothesis bigrams that match in the corresponding text, i.e., Bigram_Match=(Total number of matched bigrams in a text-hypothesis pair /Number of hypothesis bigrams).

If the value of Bigram_Match is 0.5 or more, i.e., 50% or more bigrams in the hypothesis match in the corresponding text, then the text-hypothesis pair is considered as entailment. The text-hypothesis pair is then assigned the value of 1 meaning entailment, otherwise, the pair is assigned the value of 0. The cut-off value for the Bigram_Match is based on experiments carried out on the RTE-6 Main and Novelty Task development set.

iii) Longest Common Subsequence (LCS): The Longest Common Subsequence of a text-hypothesis pair is the longest sequence of words which is common to both the text and hypothesis. LCS(T,H) estimates the similarity between text T and hypothesis H, as LCS_Match=*LCS*(*T,H*)/length of H.

If the value of LCS_Match is 0.8 or more, i.e., the length of the longest common subsequence between text T and hypothesis H is 80% or more of the length of the hypothesis , then the text-hypothesis pair is considered as entailment. The text-hypothesis pair is then assigned the value of 1 meaning entailment, otherwise, the pair is assigned a value of 0. The cut-off value for the LCS_Match is based on experiments carried out on the RTE-6 main and novelty task development set.

iv) Skip-grams: A skip-gram is any combination of n words in the order as they appear in a sentence, allowing arbitrary gaps. In the present work, only 1-skip-bigrams are considered where 1-skip-bigrams are bigrams with one word gap between two words in a sentence following the order. The measure 1-skip_bigram_Match is defined as *1_skip_bigram_Match* = *skip_gram*(*T,H*) / n, where *skip_gram*(*T,H*) refers to the number of common 1-skip-bigrams (pair of words in a sentence with one word gap) found in T and H and *n* is the number of 1-skip-bigrams in the hypothesis H.

If the value of 1_skip_bigram_Match is 0.5 or more, then the text-hypothesis pair is considered as entailment. The text-hypothesis pair is then assigned the value of 1 meaning entailment, otherwise, the pair is assigned the value of 0. The cut-off value for the 1_skip_bigram_Match is based on experiments carried out on the RTE-6 Main and Novelty Task development set.

v) Stemming: Stemming is the process of reducing terms to their root form. For example, the plural forms of a noun such as 'boxes' are transformed into 'box', and derivational endings with 'ing', 'es', 's' and 'ed' are removed from verbs. Each word in the text and hypothesis pair is stemmed using the stemming function provided along with the WordNet 2.0. If s1= number of common stemmed unigrams between text and hypothesis and s2=

number of stemmed unigrams in Hypothesis, then the measure Stemming_match is defined as Stemming_Match=s1/s2.

If the value of Stemming_Match is 0.7 or more, i.e., 70% or more stemmed unigrams in the hypothesis match in the stemmed text, then the text-hypothesis pair is considered as entailment. The text-hypothesis pair is assigned the value of 1 meaning entailment; otherwise, the pair is assigned the value of 0. The cut-off value for the Stemming_Match is based on experiments carried out on the RTE-6 Main and Novelty Task development set.

WordNet [Fellbaum, 1998] is one of most important resource for lexical analysis. The WordNet 2.0 has been used for WordNet based unigram match and stemming step. The API for WordNet Searching (JAWS)[4] is an API that provides Java applications with the ability to retrieve data from the WordNet database.

3.3 Syntactic Similarity Module

This module is based on the Stanford Parser[5], which normalizes data from the corpus of text and hypothesis pairs, accomplishes the dependency analysis and creates appropriate structures. Our Entailment system [3] uses the following features.

a) Subject: The dependency parser generates nsubj (nominal subject) and nsubjpass (passive nominal subject) tags for the subject feature. Our entailment system uses these tags.

b) Object: The dependency parser generates dobj (direct object) as object tags.

c) Verb: Verbs are wrapped with either the subject or the object.

d) Noun: The dependency parser generates nn (noun compound modifier) as noun tags.

e) Preposition: Different type of prepositional tags are prep_in, prep_to, prep_with etc. For example, in the sentence “A plane crashes in Italy.”, the identified prepositional tag is prep_in(in, Italy).

f) Determiner: Determiner denotes a relation with a noun phase. The dependency parser generates det as determiner tags. For example, the parsing of the sentence “A journalist reports on his own murders.” generates the determiner relation as det(journalist,A).

g) Number: The numeric modifier of a noun phrase is any number phrase. The dependency parser generates num (numeric modifier). For example, the parsing of the sentence “Nigeria seizes 80 tonnes of drugs.” generates the relation num (tonnes, 80).

For the sentence, “John Yoo served in the Justice Department.”, the Stanford Dependency Parser generates the following set of dependency relations:

[nn(Yoo-2, John-1), nsubj(served-3, Yoo-2), det(Department-7, the-5), nn(Department-7, Justice-6), prep_in(served-3, Department-7)]

3.3.1 Matching Module

After dependency relations are identified for both the text and the hypothesis in each pair, the hypothesis relations are compared with the text relations. The different features that are compared are noted below. In all the comparisons, a matching score of 1 is considered when the complete dependency relation along with all of its arguments match in both the text and the hypothesis. In case of a partial match for a dependency relation, a matching score of 0.5 is assumed.

a) Subject-Verb Comparison: The system compares hypothesis subject and verb with text subject and verb that are identified through the nsubj and nsubjpass dependency relations. A matching score of 1 is assigned in case of a complete match. Otherwise, the system considers the following matching process.

b) Subject-Subject Comparison: The system compares hypothesis subject with text subject. If a match is found, a score of 0.5 is assigned to the match.

c) Object-Verb Comparison: The system compares hypothesis object and verb with text object and verb that are identified through dobj dependency relation. In case of a match, a matching score of 0.5 is assigned.

d) Cross Subject-Object Comparison: The system compares hypothesis subject and verb with text object and verb or hypothesis object and verb with text subject and verb. In case of a match, a matching score of 0.5 is assigned.

e) Number Comparison: The system compares numbers along with units in the hypothesis with similar numbers along with units in the text. Units are first compared and if they match then the corresponding numbers are compared. In case of a match, a matching score of 1 is assigned.

[4] http://lyle.smu.edu/~tspell/jaws/index.html

[5] http://nlp.stanford.edu/software/lex-parser.shtml

f) Noun Comparison: The system compares hypothesis noun words with text noun words that are identified through nn dependency relation. In case of a match, a matching score of 1 is assigned.

g) Prepositional Phrase Comparison: The system compares the prepositional dependency relations in the hypothesis with the corresponding relations in the text and then checks for the noun words that are arguments of the relation. In case of a match, a matching score of 1 is assigned.

h) Determiner Comparison: The system compares the determiner in the hypothesis and in the text that are identified through det relation. In case of a match, a matching score of 1 is assigned.

j) Other relation Comparison: Besides the above relations that are compared, all other remaining relations are compared verbatim in the hypothesis and in the text. In case of a match, a matching score of 1 is assigned.

Each of the matches through the above comparisons is assigned some weight learned from the RTE-6 development corpus. A threshold of 0.30 has been set on the fraction of matching hypothesis relations based on the development set results that gives optimal precision and recall values for both YES and NO entailment. The threshold score has been applied on the RTE-6 test set using the same methods of dependency parsing followed by comparisons.

3.4 Chunking Module

In this module, we have first worked on the hypothesis side. We have extracted the part of speech (POS) tags of the hypothesis sentences using Stanford POS tagger. After getting the POS information we have extracted the chunk output using CRF Chunker [4]. Our chunk boundary detector detects each individual chunk such as noun chunk, verb chunk etc. Thus, all the chunks for each sentence in the hypothesis are identified. On the text side we have considered the specified (*.sgm) file and have extract the sentence which contain at least one noun chunk or noun word, i.e., the head word of the noun chunk. Each sentence of the text side is also processed in the same way as has been done for the hypothesis sentences.

This module contains the following sub-modules:

3.4.1 Key chunk analyzer

The key chunk analyzer identifies the key chunk in the hypothesis. We have extracted subject and object noun from the hypothesis by using Stanford dependency parser. From the dependency output we considered nsubj and nsubjpass relation for identifying subject noun and dobj relation for identifying object noun. Now we have checked each chunk for subject and object noun and consider those chunks containing subject and object noun as key chunks. In case of verb chunk we have extracted the main verb to find out the corresponding synset in the WordNet. Additional verb chunks are generated by replacing the main verb with members from its sysnet.

3.4.2 Chunk matching and scoring module

Each key chunk of the hypothesis is now searched in the text side and the sentences are extracted that contain the key chunk words. The extracted sentences are analyzed into chunks as we have done for the hypothesis.

Each individual chunk, including key chunks and generated verb chunks of the hypothesis are matched with the chunk output of the text side sentences. If chunks are matched then we give score for each individual text corresponding to the hypothesis. The scoring values are changed according to the matching of chunk and word containing the chunk. The entire scoring calculation is given below:

N= Total number of chunk containing hypothesis.
M[i]=Match score for [i]th chunk.
Wm[i]=Number of words matched in [i]th chunk.
Wc[i]=Total number of words containing the [i]th chunk.
M[i]=Wm[i] / Wc[i];

Overall score (S) = $\sum_{i=1}^{N} M[i] / N$

The score (S) will be assigned more weight by adding a constant value if it matches with a key chunk.

3.4.3 Ranking

After giving score for each text sentence corresponding to the individual hypothesis, we have ranked them according to their score and taken the best 3 ranked text sentences. Here we have considered some cutoff scores. If the ranked score is below the cutoff score then we simply discard them otherwise we have taken all the three sentences with the best rank scores.

3.5 Named Entities Module

In this module we have tagged named entities in both hypothesis and text using Stanford POS tagger. The named entities identified in the hypothesis are matched in the text file. If named entities are matched in both the sides, the text entails the hypothesis; otherwise the text does not entail the hypothesis.

3.5.1 Acronym Generator

Sometimes, multi word named entities may be present as an acronym either in the text or in the hypothesis. For every multi word named entities identified in the hypothesis, the acronym is generated by taking the first letter of each word in the named entity. The multi word named entity and its acronym forms a set and this set is compared with the named entities identified in the text. The generation of the acronym for a multi word named entity and its use in the named entity comparison process has improved the performance of the entailment decision.

3.5.2 Combined chunk-named entity module

During the matching of the key chunks and the generated verb chunks, the named entities are considered along with the acronym generated. Text sentences are assigned scores for ranking and the best three sentences according to the rank score are identified.

3.6 Features for Machine Learning

The machine learning based TE system is based on the Support Vector Machine (SVM) that uses twenty five features: lexical similarity (5 features), Lexical Distance (17 features), Chunk Module Output, NE module Output and the output tag from a rule based syntactic two-way TE system as another feature. Our system uses LIBSVM[6] to determine whether each T-H pair constitutes a correct textual entailment or not. The important lexical features that are used in the present system are: WordNet based unigram match, bigram match, longest common sub-sequence, skip-gram and stemming. In the syntactic TE system, the important features used are: subject-subject comparison, subject-verb comparison, object-verb comparison and cross subject-verb comparison. The Lexical Distance features are Block distance, Chapman length deviation, Chapman mean length, Cosine similarity, Dice similarity, Euclidean distance, Jaccard similarity, Jaro, Jaro Winkler, Matching coefficient, Monge Elkan distance, Needleman Wunch distance, Overlap Coefficient, QGrams distance, Smith-Waterman distance, Smith-Waterman with gotoh and Smith-Waterman with gotoh windowed affine.

The 17 features for lexical distance are calculated by using SimMetrics Library[7]. SimMetrics is an open source extensible library of Similarity or Distance Metrics. SimMetrics provides a library of float based similarity measures between String Data as well as the typical unnormalised metric output. We have trained our system on the RTE-6 Main Task development data and tested on the RTE-6 Main Task test data. For Novelty detection task, we have trained our system on the RTE-6 Novelty Task development data and tested on the RTE-6 Novelty Task test data.

3.7 RTE-6 KBP Validation Pilot

We developed two systems for RTE-6 KBP, one for generic task and another for the tailored task. The Apache Lucene[8] IR system has been used for the RTE-6 KBP task. Lucene follows the standard IR model with Document parsing, Document Indexing, TF-IDF calculation, query parsing and finally searching/document retrieval. Some modules in Lucene have been upgraded for our present need as described below. For TAC RTE-6 KBP 2010, the source web documents are full of noise mixed with the actual content. In that case it is very difficult to identify and separate this noise from the actual content. The corpus has much noise in the documents and the documents are in tagged format. First of all the documents have to be preprocessed. The document structure is checked and reformatted according to the system requirements. For the RTE-6 KBP generic task, we create the Query Word by the disjunction of hypothesis text after removal of the stop words along with the conjunction of the values of the "<entity>" and the "<value>" tags.

From the RTE-6 KBP (*.xml) source file, we extracted the following features for a particular pair id such as query, entity type, entity, value, attribute, text file name and a set of hypothesis.

[6] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[7] http://staffwww.dcs.shef.ac.uk/people/S.Chapman/simmetrics.html
[8] http://lucene.apache.org/

i. Generic Task

Method 1:

RTE-6 KBP Lexical engine: In the lexical engine we have passed the entity, value and attribute information and the corresponding set of hypothesis and text files. If the entity type is a named entity such as person, organization etc in Table 2, the generated acronym of the named entity is also provided as an input. If the entity or its acronym as well as the value are found in the sentences of the text file then these sentences are further considered. If no match is found in any of the text file sentences, no further processing is done and the text file is considered as not entailing the hypothesis. For those text file sentences where a match of the entity and value are found, the main verbs in the hypothesis sentences are identified. In the corresponding text file we have extracted the list of all the main verbs in between the matched entity and value. The verb of the hypothesis side and the verb of the text side are checked to see whether they belong to the same hypernym tree or are members of the same synset. If so then an entailment decision of “YES” is taken, otherwise the entailment decision is “NO”.

Method 2:

At first we parsed the (*.sgm) files. After parsing, the documents are indexed using Lucene, an open source full text search engine. The basic architecture of Lucene is shown in Figure 1.

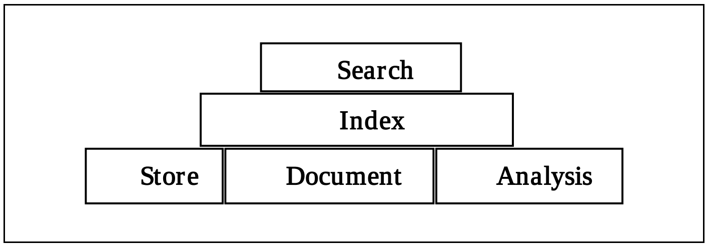

Figure 1. Lucene Architecture

After indexing has been done, the queries have to be fired to retrieve relevant documents. We take only the top ranked retrieved document assuming that it is the most relevant document for the query generated from the hypothesis. If the text file corresponding to a hypothesis is included in the most relevant document retrieved by Lucene, then the text-hypothesis pair is considered to have entailed, otherwise there is no entailment.

ii. Tailored Task:

For the tailored task, we developed the validation rules for each attribute from the development data. At first we have identified the entity and value in each hypothesis using the RASP NER [4]. Some RASP Named Entity tags are shown in the following Table 2.

	NE Tag	NE Tag Example
Number	<phr c="cd">	<phr c="cd">one</phr>
Person Name	<enamex type="person">	<enamex type="person">Chris Simcox</enamex>
Organization Name	<enamex type="organization">	<enamex type="organization">Minuteman Civil Defense Corps</enamex>
Location Name	<enamex type="location">	<enamex type="location">Argentina</enamex>
Role/ Title	<role>	<role>President</role>
Date/ Time	<timex type="date">	<timex type="date">2004</timex>

Table 2. RASP NE Table

Some of validation rules are described as follows:

Attribute:: title

For attribute *title*, we developed a list of possible titles using the development set data and the Wikipedia. Then we check the value of <value> tag of this pair id, looking for a word match with the title list. If there is a match then the entailment value for the pair is evaluated as ”YES” otherwise “NO”.

Attribute:: city of birth

At first we check whether the value of <value> tag is a location or not based on the RASP NE output. We have developed a list of city names of various countries of the world using the list available on the web[9]. Then value of <value> tag is compared with the corresponding database file. If a match is found then we look for phrases similar to “born in” in the text. If a match is found then the entailment value for the pair is "YES" otherwise "NO".

Attribute:: city of death

At first we check whether the value of <value> tag is location or not based on the RASP NE output. We have compared the value of <value> tag with the list of city names as developed. If a match is found then we look for phrases similar to "passed away " in the text. If a match is found then the entailment value for the pair is "YES" otherwise "NO".

Attribute:: website

At first we check whether the value of <value> tag is url or not based on the RASP NE output. If it is an url then the entailment value for the pair is "YES" otherwise "NO".

Attribute:: cause_of_death

At first we make a list of cause of diseases and a causal verbs list using information available in the WordNet and the development set. Then we compare the value of the <value> tag with this causal list. If a match is found then the entailment value for the pair is "YES" otherwise "NO".

Attribute::members

For this attribute, we check whether the value of <value> tag is organization or not based on the RASP NE output. If the <value> tag is an organization, then the entailment value for the pair is "YES" otherwise "NO".

For named entity recognition, the RASP Parser (Briscoe et al., 2006) nertag component has been used. The nertag component is a rule-based named entity recognizer which recognizes and marks up the following kinds of named entity: numex (sums of money and percentages), timex (dates and times) and enamex (persons, organizations and locations).

4. Experiments on the Development and the Test data and the results

4.1 Main Task

For the Main Task development and test set we have prepared the three runs which are as follows:

Run 1 (JU_CSE_TAC1_Main): Lexical and Syntactic Entailment.

Run 2 (JU_CSE_TAC2_Main): Lexical, Chunk, Named Entities and Syntactic Entailment.

Run 3 (JU_CSE_TAC3_Main): Using SVM Machine Learning (Features: Lexical Distance, Lexical Entailment, Chunk, Named Entities and Syntactic Entailment).

The results of the Main Task on the development set are shown in Table 3.

Run Id#	Precision	Recall	F-Score
1	28.07	26.53	27.28
2	53.76	20.74	29.93

Table 3: Micro Average Result on RTE-6 Main Development Set

The results of the Main Task on the Test set are shown in Table 4.

Run Id#	Precision	Recall	F-Score
1	38.63	31.64	34.79
2	38.49	20.53	26.78
3	78.30	19.47	31.19

Table 4: Micro Average Result on RTE-6 Main Test Set

4.2 Novelty Task

[9] http://www.maxmind.com/app/worldcities

For the Novelty Task development and test set, we have prepared the three runs which are as follows:
Run 1 (JU_CSE_TAC1_Novelty): Lexical and Syntactic Entailment.
Run 2 (JU_CSE_TAC2_Novelty): Lexical, Chunk, Named Entities and Syntactic Entailment.
Run 3 (JU_CSE_TAC3_Novelty): Using SVM Machine Learning (Features: Lexical Distance, Lexical Entailment, Chunk, Named Entities and Syntactic Entailment).

The results of the Novelty Task on the Development set are shown in Table 6.

Run Id#	Evaluation	Precision	Recall	F-Score
1	**Novelty Evaluation**	77.91	75.28	76.57
	Justification Evaluation	30.93	24.89	27.59
2	**Novelty Evaluation**	67.65	77.53	72.25
	Justification Evaluation	35.03	7.78	12.73

Table 6: Micro Average Result on RTE-6 Main Development Set

The results of the Novelty Task on the test set are shown in Table 7.

Run Id#	Evaluation	Precision	Recall	F-Score
1	**Novelty Evaluation**	80.58	83.00	81.77
	Justification Evaluation	40.92	29.60	34.35
2	**Novelty Evaluation**	71.67	86.00	78.18
	Justification Evaluation	41.26	19.92	26.87
3	**Novelty Evaluation**	66.67	96.00	78.69
	Justification Evaluation	75.71	14.66	24.57

Table 7: Micro Average Result on RTE-6 Novelty Test Set

4.3 RTE-6 KBP Validation Pilot Task

4.3.1 Generic Task

Run 1 (JU_CSE_TAC1_general): Using Apache Lucene (Using Method 2, Section 3.7)
Run 2 (JU_CSE_TAC2_general): Manual Generated Rules and Apache Lucene (Using Method 2, Section 3.7)
Run 3 (JU_CSE_TAC3_general): Lexical Entailment (Using Method 1, Section 3.7)

The results of the KBP Validation Task on the Test set for generic task are shown in Table 8.

Run Id#	Precision	Recall	F-Score
1	0.0925	0.2906	0.1403
2	0.224	0.1396	0.172
3	0.0991	0.3368	0.1531

Table 8: Micro Average Result on RTE-6 KBP Validation Test Set (Generic Task)

4.3.2 Tailored Task

Run 1 (JU_CSE_TAC1_tailored): Use only Manual Generated Rules (Section 3.7)
Run 2 (JU_CSE_TAC2_tailored): Checking the Document and Manual Generated Rules (Section 3.7)

Run 3 (JU_CSE_TAC3_tailored): Checking the Document and Fine tuned Manual Generated Rules (Section 3.7)
The results of the KBP Validation Task on the Test set for tailored are shown in Table 9.

Run Id#	Precision	Recall	F-Score
1	0.2218	0.4636	0.3
2	0.2432	0.5167	0.3307
3	0.2424	0.5108	0.3288

Table 9: Micro Average Result on RTE-6 KBP Validation Test Set (Tailored Task)

5. Ablations test and results

An ablation test [6] consists of removing one module at a time from a system, and rerunning the system on the test set with the other modules, except the one tested. Comparing the results to those obtained by the system as a whole, it is possible to assess the practical contribution of each single module. In order to better understand the relevance of the knowledge resources used by RTE systems and evaluate the contribution of each of them to the systems' performances, ablation tests for major knowledge resources are required for such systems.

For Main task, we have used the WordNet as a resource. So in Run 1, Run 2 and Run 3, we have ablated the WordNet resource.
For Ablation Test,
<run> attributes: The Micro-Average scores obtained in the Main task submission;
<ablation> attributes: The Micro-Average scores obtained in the ablation test;
<resource_impact> attributes: The difference between the Main task scores and the ablation test scores, measuring the impact of the ablated resource. Positive values indicate that the removed resource/tool has a positive impact on the performance of the system, as the results worsen after its removal. Negative values indicate the contrary. Table 10 shows the results of the ablation test for the main task with the WordNet resource being ablated.

Run Id#	Run Description	Precision	Recall	F-Score
1	**<run>**	38.63	31.64	34.79
	<ablation>	86.57	12.28	21.50
	<resource_impact>	-47.94	19.36	13.29
2	**<run>**	38.49	20.53	26.78
	<ablation>	83.65	9.21	16.59
	<resource_impact>	-45.16	11.32	10.19
3	**<run>**	78.30	19.47	31.19
	<ablation>	76.96	16.61	27.33
	<resource_impact>	1.34	2.86	3.86

Table 10: Ablation Test for main task.

6. Conclusion

The textual entailment system has been developed as part of the participation in the TAC 2010 Recognizing Textual Entailment (RTE) Track organized by National Institute of Standards and Technology (NIST). We have proposed a textual entailment recognition system framework which is a combination of lexical, syntactic and semantic features. The overall system has been evaluated using the evaluation metrics provided as part of the TAC RTE 2010 track. The evaluation results are satisfactory considering that this is the second participation in the track. Future works will be motivated towards improving the performance of the system. It has been observed that the anaphora and coreference resolution in the text followed by comparison with the hypothesis would have improved the performance of the textual entailment system.

References:

1. Luisa Bentivogli, Peter Clark, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo. "The Sixth PASCAL Recognizing Textual Entailment Challenge". In TAC 2010 Workshop Notebook, Maryland, USA.
2. Partha Pakray, Sivaji Bandyopadhyay, Alexander Gelbukh, "Lexical based two-way RTE System at RTE-5", System Report, TAC RTE Notebook, 2009.
3. Xuan-Hieu Phan, "CRFChunker: CRF English Phrase Chunker". In PACLIC 2006. (2006)
4. Partha Pakray, Alexander Gelbukh and Sivaji Bandyopadhyay, "A Syntactic Textual Entailment System Using Dependency Parser", Springer Berlin / Heidelberg, Volume Volume 6008/2010, Book Computational Linguistics and Intelligent Text Processing, ISBN 978-3-642-12115-9, Pages 269-278.
5. E. Briscoe, J. Carroll, and R. Watson: The Second Release of the RASP System. In Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions.
6. Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, Bernardo Magnini. The Fifth PASCAL Recognizing Textual Entailment Challenge. In TAC 2009 Workshop Notebook, Gai-thersburg, Maryland, USA.