A Statistical Approach with Syntactic and Semantic Features for Chinese Textual Entailment

Chun Tu [1], Min-Yuh Day [2]

[1] *Department of Information Management, Tamkang University, Taiwan*
[2] *Department of Information Management, Tamkang University, Taiwan*

kevincncod2@gmail.com, myday@mail.tku.edu.tw

Abstract

Recognizing Textual Entailment (RTE) is a PASCAL/TAC task in which two text fragments are processed by system to determine whether the meaning of hypothesis is entailed from another text or not. In this paper, we proposed a textual entailment system using a statistical approach that integrates syntactic and semantic techniques for Recognizing Inference in Text (RITE) using the NTCIR-9 RITE task and make a comparison between semantic and syntactic features based on their differences. We thoroughly evaluate our approach using subtasks of the NTCIR-9 RITE. As a result, our system achieved 73.28% accuracy on the Chinese Binary-Class (BC) subtask with NTCIR-9 RITE. Thorough experiments with the text fragments provided by the NTCIR-9 RITE task show that the proposed approach can significantly improve system accuracy.

Keywords: Textual Entailment, Semantic Features, Syntactic Features, Machine Learning, Support Vector Machine (SVM)

1. INTRODUCTION

Recognizing Textual Entailment (RTE) is a task in which a system is given two text fragments and then determine whether the meaning of hypothesis is entailed from another text [1]. There are two subtasks in RTE: RTE 2-way and RTE 3-way. RTE 2-way output yields two labels: "Entailment" and "No Entailment". The label "Entailment" is given when the Text entails the Hypothesis, while "No Entailment" is given when the Text does not entail the Hypothesis. RTE 3-way output gives three labels: "Entailment" Text entails Hypothesis), "Contradiction" (Text contradicts Hypothesis), and "Unknown". (relationship between Text and Hypothesis is unknown) [2].

Since 2005, the importance of RTE in assessing semantic inference in text has been increasing. After the third PASCAL RTE Challenges in Europe, RTE became one of tasks of the Text Analysis Conference (TAC) in 2008. The RTE Challenge is a generic task that captures major semantic inference needs across many natural language processing applications, such as Question Answering (QA), Information Retrieval (IR), Information Extraction (IE), and (multi) document summarization. RTE is largely European and American project. Its counterpart in East Asia is called, Recognizing Inference in Text (RITE) [3]. RITE is a generic benchmark task that addresses major text understanding needs in variety of NLP/Information Access research areas. There are three subtasks in RITE: Binary-Class (BC), Multi-Class (MC), and RITE4QA. In all subtasks, an input is two text fragments while the output is one of two or five labels. In the BC subtask, there are two output labels: "Yes" and "No". In the MC subtask, there are five output labels: "Forward", "Reverse", "Bidirection", "Contradiction" and "Independence". In the RITE4QA subtask, the input and output are identical to the BC subtask ,but as an embedded answer validation component in Question Answering system [3]. For instance, in the BC subtask, an input text appears as follows:

T1: 香港的主權和領土是在 1997 由英國歸還給中國的。
(Hong Kong's sovereignty and territories were returned to China by the United Kingdom in 1997)
T2: 1997 年香港回歸中國。
(Hong Kong was returned to China in 1997)

The system output for the BC subtask is "YES" for the above T1, T2 pair. Furthermore, the system output for the MC subtask is "Forward" where T2 can be inferred from T1, but T1 cannot be inferred from T2. Here is another instance of the MC subtask:

T1: 尼泊爾毛派叛亂份子攻擊安全警衛哨站。
(Nepal's Maoist insurgents assaulted a security guard outpost)
T2: 尼泊爾毛派游擊隊攻擊民航機。
(Nepal's Maoist guerrillas assaulted civil aviation aircraft)

The system output for the BC subtask is "NO" for the above T1, T2 pair. Further, the system output for the MC subtask is "Contradiction" where either T1 cannot be inferred from T2 or T2 cannot be inferred from T1.

Generally, features used for dealing with TE can be roughly divided into two categories, syntactic features and semantic features Semantic features include synonyms, antonyms, and negation. Most studies emphasize semantic features in text fragments. For example:

978-1-4673-2284-3/12/$31.00 ©2012 IEEE

T1: 車諾比病毒在1999年4月總共造成超過200萬台電腦無法開機

(CIH caused severe boot problems in more than 200 million computers in April, 1999)

T2: 1999年4月車諾比病毒總共造成逾200萬台電腦無法開機

(CIH caused severe boot problems in over 200 million computers in April, 1999)

If we consider only syntactic features, the output would be "Forward". However, if we consider both syntactic features and semantic features "超過(more than)" and "逾(over)" are synonyms. Therefore, the output would be "Bidirection" which is the correct answer.

In sum, semantic features and syntactic features are comprehensively discussed in most studies. Using syntactic approaches, we can simply relate two text fragments from their sentence string lengths, though this results in biases in the outputs. Therefore, it is necessary to consider semantic features in text entailment. In this paper, we propose a novel system, aimed at enhancing the accuracy of a model and compare the effectiveness and efficiency of semantic and syntactic feature processing on the RITE subtasks.

The remainder of this paper is organized as follows. Section 2 describes the literature on Recognizing Textual Entailment and machine learning. Section 3 details our system framework and the features we adopted. Section 4 shows the experimental setup and the evaluation of our approach. Finally, Section 5 presents our conclusions.

2. LITERATURE REVIEW

In this section, we provide the research background on RTE, RITE and machine learning with related approaches to this problem. We then lay the foundation for our proposed approach by reviewing the literature on the use of these approaches.

2.1. English Textual Entailment

RTE mainly uses two sets of features, semantic features and syntactic features. Siblini and Kosseim [4] proposed a Ontology Alignment System (OAS) which adopted syntactic features and semantic features with ontology alignment and acquisition to deal with Text and Hypothesis, respectively. However, the application of OAS is limited by cognitive differences the in text fragments. For example, "bank" might have different meanings in different contexts. In terms of finance, “bank” means “銀行” while in terms of ecology, “bank” means “河岸”, which may result in problems when dealing with semantic features.

Burchardt et al. [5] proposed the SALSA system which adopted semantic features for inference analysis of text fragments. They offered a match graph for synonym words. They used 47 features to calculate the similarity in each graph for training. However, this approach encounters the same problem as that of Siblini and Kosseim above: when a word appears in different contexts, it may have different meanings. Bias would thus occur when training with these datasets.

Iftene and Moruz [6] proposed an approach that used Positive cases, Negative cases, Contradiction cases and Unknown cases (words are not decisive in determining the type of entailment) to analyze the inference of two text fragments. Vanderwende et al. [7], concluded that nearly 48% of text fragments could be inferred merely by syntactic features plus a general-purpose thesaurus. Castillo [8] proposed an approach using Edit Distance and Longest Common Substring (LCS) to recognize the inference of text fragments. Kouylekov and Magnini [9] proposed a Tree Edit Distance approach to analyze the similarity of text fragments.

In sum, compared to Chinese, when we process English text fragments, each word is split explicitly in an English sentence and much information is carried by the use of auxiliaries and by verb inflections. Chinese, on the other hand, is an uninflected language and conveys meaning through word order, adverbials or shared understanding of the context.

2.2. Chinese Textual Entailment

An issue for RITE is that Chinese and Japanese are relatively more complicated than English for text inference. Therefore, understanding the subtle differences in Chinese and Japanese is harder. Zhang and Yamamoto [10] proposed a pattern-based approach to paraphrase the text fragments in which the meaning is retained to the greatest possible extent without deep parsing. Therefore, some words may have slightly different meanings when rendered in Simplified Chinese and Traditional Chinese. Li et al. [11] proposed a Chinese Characters Conversion System to tackle word ambiguity both in Simplified and Traditional Chinese.

Thus, text fragments in Chinese or Japanese are slightly different than English text fragments. For example, “haste” in Chinese has several meanings: "迅速","急躁". The former word has a positive meaning while the latter word has a negative meaning. Therefore, Chinese might encounter one-to-many ambiguity problem.

2.3. Machine Learning

Malakasiotis and Androutsopoulos [12] proposed a Support Vector Machine (SVM) approach with semantic features to tackle text fragments and train a model which contains 128 features in order to increase its accuracy.

In sum, considering SVM as machine learning tool [13] which can solve classification and clustering problem within feasible time limits as well as select the best feature combinations to enhance model accuracy and model efficiency.

3. SYSTEM ARCHITECTURE

We developed a textual entailment system using a hybrid approach that integrates syntactic features, semantic features and machine learning techniques for recognizing inference in text in an NTCIR-9 RITE task. Figure 1 shows the proposed system architecture of the IMTKU Textual Entailment System for Recognizing Inference in Text in an NTCIR-9 RITE task.

First, we preprocess XML training datasets. Preprocess include two steps, text pair extraction and segmentation. We extract each text pair from the XML training datasets, and then tokenize each text pair into words and phrases for analysis. After preprocessing, we designate a variety of features for training. For semantic features, we adopt TYCCL[14] from HIT University. We do a format conversion to the TYCCL in order to analyze the text inference more easily. We then train a model using an SVM. After several adjustments (ex: Cross-Validation), the model can be used to predict test datasets.

3.1. Preprocessing

We extracted text fragments from NTCIR-9 RITE RITE4QA raw datasets and use CKIP Autotag[15] for producing available datasets.

3.1.1. XML dataset extraction

We extracted IDs and text pairs from raw XML datasets of the RITE corpus for analysis.

3.1.2. Data format unification

A word may be expressed in different ways. For example, 1990 may be written "1990 年" or "一九九零年". It is thus necessary to unify the data format. [16]

3.1.3. CKIP Autotag

We adopt the Chinese Knowledge and Information Processing (CKIP) System to process text pairs for analysis.

3.2. Feature Generation

We designated 14 semantic and syntactic features:

Word Similarity, String Length, String Length Difference, String Length Ratio, Longest Common Substring (LCS), Char-Based Edit Distance, Word Length, Word Length Difference, Word Length Ratio, Word-Based Edit Distance.

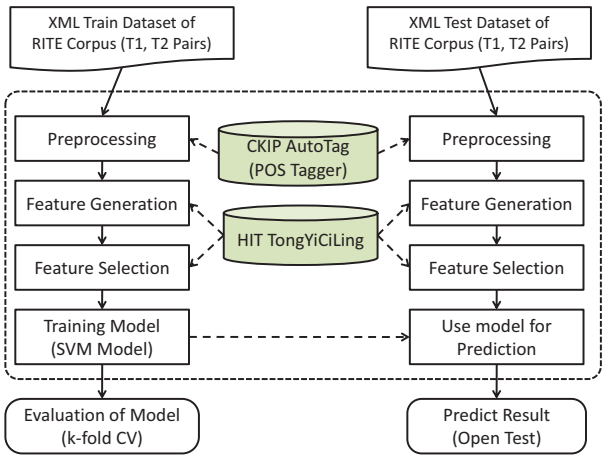

Figure 1. System Architecture of IMTKU Text Entailment System in NTCIR-9 RITE

(1) String Length/Length Difference/Ratio

Basic syntactic approach we adopted as a feature. We use string length difference as a feature to reduce bias on a length basis. We can use string length ratio to confine the range between 0 and 1 to reduce bias and enhance accuracy.

(2) Longest Common Substring

We use Longest Common Substring [17] to find similarity in text pairs. The formula is:

$$LCS(\mathbf{X}_{1\ldots i},\mathbf{Y}_{1\ldots j})=\begin{cases}\emptyset & \text{If i=0 or j=0}\\ LCS(X_{1\ldots i-1},Y_{1\ldots j-1})+x_i & \text{If } x_i = y_i\\ \max\left(LCS(X_{1\ldots i},Y_{1\ldots j-1}),LCS(X_{1\ldots i-1},Y_{1\ldots j})\right) & \text{else}\end{cases}$$

the formula finds the longest string (or strings) that is a substring (or are substrings) of two or more strings. We first find the longest subsequences common to X_i and Y_i and then compare the elements x_i and y_i. If they are equal, then the sequence LCS (X_{i-1}, Y_{i-1}) is extended by that element, x_i. If they are not equal, then the longer of the two sequences, LCS (X_i, Y_{i-1}), and LCS (X_{i-1}, Y_i), is retained (if they are both the same length, but not identical, then both are retained.) Notice that the subscripts are reduced by 1 in these formulas, which can result in a subscript of 0. Since the sequence elements are defined to start at 1, it was necessary to add the requirement that the LCS is empty when a subscript is zero.

(3) Char-based Edit Distance

Edit Distance is a distance in which insertions and deletions have equal cost and replacements have twice the cost of an insertion. It is thus the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character. For instance:

T1: 我喜歡打籃球 (I like to play basketball)

T2: 我討厭打籃球 (I hate to play basketball)

In the text pair, the edit distance is 2 since the character "喜" undergoes one replacement, becoming into "討", while "歡" undergoes one replacement to become into "厭"

(4) Word Length/Difference/Ratio

We use CKIP Autotag to tokenize sentences into every word and calculate the total words. We use string word length difference as a feature to reduce bias on a word length basis. We can use word length ratio to confine a range between 0 and 1. In other words, the word length ratio is used to reduce bias and enhance accuracy.

(5) Word-based Edit Distance

Edit Distance is to measure distance as the number of operations required to transform a string into another where this feature is token-based. For instance:

T1: 我 (I)(N) 喜 歡 (Like)(Vt) 打 (to play)(Vt) 球 (basketball)(N)

T2: 我 (I)(N) 討 厭 (hate)(Vt) 打 (to play)(Vt) 球 (basketball)(N)

In this text pair, the edit distance is 1 where the word "喜歡"(like) transforms into "討厭"(hate).

(6) Noun/Verb Number

We incorporated a feature which calculates noun/verb numbers in a sentence, so we could do a simple comparison in advance.

(7) Word Semantic (Synonym) Similarity

We proposed a semantic feature that uses HIT TYCCL where each word in the TYCCL is assigned an ID and words with same ID are considered synonyms. For example:

Di01A01=世界, 世, 世上, 大地, 天下, 天底下, 全世界, 環球, 全球, 舉世, 中外, 寰宇, 五洲, 海內, 海內外, 五湖四海, 大千世界, 大世界, 普天之下

However, using the original TYCCL for recognizing texts may be too complicated because each synonym has its own ID number, meaning that the more synonyms a word has, the more complicated the queries are. Thus, data may be hard to maintain and update because those synonyms are correlated. Therefore, we do a format conversion to the TYCCL and also added a similarity value for querying.

Formula: TYCCL Scoring Function: $((\tau-\rho) + 1) / \tau$

τ:synonym number ρ: word ranking in synonym list

For example, 世界(World) has 19 synonyms. The synonym list shows that the word 世界 (World) has the highest ranking in the 世界(World) synonym list, so we calculate its similarity score as

$$(19-1)+1/19 = 19/19 = 1$$

Table 1 Comparison of non-semantic features with semantic features

Non-semantic features output	**With semantic features output**
Result: **Forward**	Result: **Binary**
T1 String Length:30	T1 String Length:30
T2 String Length:28	T2 String Length:28
T1_T2 length Difference:2	T1_T2 length Difference:2
T1_T2 ratio:1.0714	T1_T2 ratio:1.0714
LCS:22	LCS:22
T1 Word Length:13	T1 Word Length:13
T2 Word Length:12	T2 Word Length:12
T1_T2 Word Length Ratio:1.083	T1_T2 Word Length Ratio:1.083
T1_T2 Word Length Difference:1	T1_T2 Word Length Difference:1
Edit Difference: 13	Edit Difference: 13
Edit Word Distance: 6	Edit Word Distance: 6
Noun Number Difference: 0	Noun Number Difference: 0
Verb Number Difference: 0	Verb Number Difference: 0
	Word Semantic (Synonym) Similarity: 12.6042

Thus, the word 世界 (World) has a similarity of 1 in the 世界 (World) synonym list, meaning that it is 100% similar. After calculating word similarity, the results are shown as follows:

世界 Di01A01=| 世 界 :1.0000, Di14C04=| 世風:0.5000, Dd05B03=|領域:0.3333

The results showed the list of synonyms of the word 世界. Each synonym has its ID and its similarity value to 世界.

The results show that if we compare 世界 and 世風 on a syntactic basis, they as appear to be two independent words, but on a semantic basis, 世風 is 50% similar to 世界, which could decrease the experimental bias.

We can also evaluate text fragments via word similarity. We use CKIP Autotag on each text fragments in order to calculate their similarities on a word basis, not on a char basis, and reduce experimental bias. For example:

T1: 卓諾比病毒在1999年4月總共造成超過200萬台電腦無法開機

(CIH caused severe boot problems in more than 200 million computers in April, 1999)

T2: 1999年4月卓諾比病毒總共造成逾200萬台電腦無法開機

(CIH caused severe boot problems over in 200 million computers in April, 1999)

Table 1 show that if we consider only syntactic features, the output would be "Forward" because the T1 String Length is longer than the T2 String Length. However, if we consider semantic features, the output would be "Binary" because the word 超過 (more than) and 逾(over) are synonyms.

3.3 Machine Learning

We used LibSVM as the machine learning module. [13] LibSVM provides two tools for enhancing model accuracy: grid.py and fselect.py. These two tools select the best parameters and best features for the model.

4. EXPERIMENTAL RESULTS AND ANALYSIS

We use the RITE1 CT MC Development set of 421 training pairs provided by NTCIR-9 RITE and the RITE1 CT MC Development set of 900 test pairs provided by NTCIR-9 RITE for prediction. Table 2 shows the experimental results of the Cross Validation (CV) and Open Test outputs for each feature.

Table 2 showed that features with difference outperformed other features. The accuracy nearly reached 70%, suggesting that text pair lengths are key factors in text inference.

Table 3 and 4 showed that config 1 to 3 performed better than config 4 to 6 in Cross Validation and the experimental groups outperformed control groups on the Open Test.

In sum, semantic features do work well when combined with other features. As Table 2 and 3 shows, config with semantic features performs better than other configs without semantic features on the Open Test. Adding Word Semantic Similarity to the model increases the accuracy to 70%.

It is necessary to select appropriate parameters in training. We used Grid.py to select the best parameters because different parameters influence model accuracy. Different feature combinations will also result in different models. We adopted fselect.py to select the best feature combinations in order to enhance model accuracy.

Table 5 shows that accuracy was enhanced after using grid.py and fselect.py. Accuracy without semantic features rose from 68.67% to 72.65% while accuracy with semantic features increased from 71.89% to 73.28%.

In sum, there is a significant improvement in accuracy after using grid.py and fselect.py, while configs with semantic features performed better than configs without semantic features.

Table 2 Cross Validation and Open Test results for each feature

Feature ID	Feature	Cross Validation (BC)	Open Test(BC)
Feature01	T1 String Length	58.43%	60.11%
Feature02	T2 String Length	64.13%	61.44%
Feature03	String Length Difference	68.65%	**68.56%**
Feature04	String Length Ratio	69.12%	67.11%
Feature05	Longest Common Substring	59.86%	60.67%
Feature06	Char-Based Edit Distance	60.09%	60.00%
Feature07	T1 Word Length	58.91%	60.44%
Feature08	T2 Word Length	62.93%	61.67%
Feature09	Word Length Difference	**69.12%**	67.44%
Feature10	Word Length Ratio	66.27%	66.67%
Feature11	Word-Based Edit Distance	57.48%	60.00%
Feature12	Noun Number	65.30%	66.78%
Feature13	Verb Number	58.43%	63.44%
Feature14	Word Semantic Similarity	59.95%	60.00%

Table 3 Cross Validation and Open Test results after features selection (without semantic features)

Config	Feature	Cross Validation (BC)	Open Test (BC)
Config1	Feature1~13	**73.63%**	67.78%
Config2	Feature 4~13 (except 10)	72.21%	**68.67%**
Config3	Feature 1~11	72.92%	67.11%

Table 4 Cross Validation and Open Test results after features selection (with semantic features)

Config	Feature	Cross Validation (BC)	Open Test (BC)
Config4	Feature1~14	70.02%	70.33%
Config5	Feature 4~14 (except 10)	69.64%	69.78%
Config6	Feature3,6,9,10,14	**71.23%**	**71.89%**

5. CONCLUSIONS

In this paper, we propose a novel system using both semantic and syntactic features for performing a RITE subtask. The results showed that with semantic features and machine learning methods, our methods achieved 73.28% accuracy on the Chinese Binary-Class (BC) subtask of the NTCIR-9 RITE task.

The contributions of this paper include:

(1) Compared to using only syntactic features, semantic features can enhance the accuracy of the model.

(2) Adding Support Vector Machine (SVM) machine learning tools with a training model and adjusting parameters and selecting features greatly improve the accuracy of our system.

(3) We thoroughly evaluate our approach in the context of the subtasks of the NTCIR-9 RITE. The results of our system attest the effectiveness of the approaches we propose for the NTICR-9 RITE and its subtasks.

Table 5 CV and Open Test results after using grid.py and fselect.py

Feature	Cross Validation (BC)	Open Test (BC)
Feature 4~13(Except10) (without semantic feature)	**75.29%**	72.65%
Feature 3,6,9,10,14 (with semantic feature)	73.20%	**73.28%**

6. ACKNOWLEDGEMENT

This research was supported in part by the National Science Council of Taiwan under Grants NSC 101-3113-P-032-001 and TKU research grant. We would like to thank the support of IASL, IIS, Academia Sinica, Taiwan.

7. REFERENCES

[1] H. Shima, *et al.*, "Overview of NTCIR-9 RITE: Recognizing Inference in TExt," presented at the Proceedings of NTCIR-8 Workshop Meeting, Tokyo, Japan, 2011.

[2] (2011.10.27). *Text Analysis Conference*. Available: http://www.nist.gov/tac/2010/RTE/RTE6_Main_NoveltyDetection_Task_Guidelines.pdf

[3] (2011). *NTCIR RITE*. Available: http://artigas.lti.cs.cmu.edu/rite/Main_Page_(TC)

[4] R. Siblini and L. Kosseim, "Using Ontology Alignment for TAC RTE Challenge," presented at the Proceedings of the Text Analysis Conference, Gaithersburg, MD, 2008.

[5] A. Burchardt, *et al.*, "A semantic approach to textual entailment: System evaluation and task analysis.," presented at the Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, Czech Republic, 2007.

[6] A. Iftene and M. Moruz, "UAIC participation at RTE5," presented at the Proceedings of the Text Analysis Conference, Gaithersburg, MD., 2008.

[7] L. Vanderwende, *et al.*, "What Syntax can Contribute in Entailment Tas," *Microsoft Research,* 2006.

[8] J. Castillo, J., "A Machine Learning Approach for Recognizing Textual Entailment in Spainish," 2010.

[9] M. Kouylekov and B. Magnini, "Recognizing textual entailment with tree edit distance algorithms.," presented at the Proceedings of the PASCAL Recognizing Textual Entailment Challenge, 2005.

[10] Y. Zhang and K. Yamamoto, "Paraphrasing spoken Chinese using a paraphrase corpus," *The Journal of Natural Language Engineering,* vol. 11 No.4, December 2005.

[11] M. H. Li, *et al.*, "Chinese Characters Conversion System based on Lookup Table and Language Model," presented at the Proceedings of the 22nd Conference on Computational Linguistics and Speech Processing, Nantou,Taiwan, 2010.

[12] P. Malakasiotis and I. Androutsopoulos, "Learning textual entailment using SVMs and string similarity measures.," presented at the Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague. , 2007.

[13] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol* vol. 2, pp. 1-27, 2011.

[14] J.-J. Mei, *et al.*, *TongYiCi CiLin (Chinese Synonym Forest): Shanghai Press of Lexicon and Books*, 1983.

[15] (2011.10.27). *CKIP AutoTag*. Available: http://ckipsvr.iis.sinica.edu.tw/

[16] W.-C. Huang and S.-H. Wu, "Feature Analysis of Chinese Textural Entailment Systems," presented at the Proceedings of the 23rd Conference on Computational Linguistics and Speech Processing 2011.

[17] D. S. Hirschberg, "Algorithms for the Longest Common Subsequence Problem," *Journal of the Assocrauon for Computing Machinery,* vol. 24:4, pp. 664-675, 1997.