

# A Twin-Candidate Model of Coreference Resolution with Non-Anaphor Identification Capability

Xiaofeng Yang<sup>1,2</sup>, Jian Su<sup>1</sup>, and Chew Lim Tan<sup>2</sup>

<sup>1</sup> Institute for Infocomm Research,  
21, Heng Mui Keng Terrace, Singapore, 119613  
{xiaofengy, sujian}@i2r.a-star.edu.sg

<sup>2</sup> Department of Computer Science,  
National University of Singapore, Singapore, 117543  
{yangxiao, tanc1}@comp.nus.edu.sg

**Abstract.** Although effective for antecedent determination, the traditional twin-candidate model can not prevent the invalid resolution of non-anaphors without additional measures. In this paper we propose a modified learning framework for the twin-candidate model. In the new framework, we make use of non-anaphors to create a special class of training instances, which leads to a classifier capable of identifying the cases of non-anaphors during resolution. In this way, the twin-candidate model itself could avoid the resolution of non-anaphors, and thus could be directly deployed to coreference resolution. The evaluation done on newswire domain shows that the twin-candidate based system with our modified framework achieves better and more reliable performance than those with other solutions.

## 1 Introduction

In recent years supervised learning approaches have been widely used in coreference resolution task and achieved considerable success [1,2,3,4,5]. Most of these approaches adopt the single-candidate learning model, in which coreference relation is determined between a possible anaphor and one individual candidate at a time [1,3,4]. However, it has been claimed that the reference between an anaphor and its candidate is often subject to the other competing candidates [5]. Such information is nevertheless difficult to be captured in the single-candidate model. As an alternative, several researchers proposed a twin-candidate model [2,5,6]. Instead of directly determining coreference relations, this model would judge the preference between candidates and then select the most preferred one as the antecedent. The previous work has reported that such a model can effectively help antecedent determination for anaphors [5,6].

However, one problem exists with the twin-candidate model. For every encountered NP during resolution, the model would always pick out a “best” candidate as the antecedent, even if the current NP is not an anaphor. The twin-candidate

model itself could not identify and block such invalid resolution of non-anaphors. Therefore, to apply such a model to coreference resolution, some additional efforts have to be required, e.g., using an anaphoricity determination module to eliminate non-anaphors in advance [5], or using threshold to prevent the selection of a candidate if the confidence it wins other competitors is low [6].

In this paper, we explore how to effectively apply the twin-candidate model to the coreference resolution task. We propose a modified learning framework with the capability of processing non-anaphors. In the framework, we make use of non-anaphors to create training instances. This special class of instances would enable the learned classifier to identify the test instances formed by non-anaphors during resolution. Thus, the resulting model could avoid resolving a non-anaphor to a non-existent antecedent by itself, without specifying a threshold or using an additional anaphoricity determination module. Our experiments on MUC data set systematically evaluated effectiveness of our modified learning framework. We found that with this new framework, the twin-candidate based system could not only outperform the single-candidate based one, but also achieve better and more reliable results than those twin-candidate based systems using the two mentioned solutions.

The rest of the paper is organized as follows. Section 2 describes the original framework of the twin-candidate model. Section 3 presents in details the modified framework, including the training and resolution procedures. Section 4 reports and discusses the experimental results and finally Section 6 gives the conclusions.

## 2 The Original Framework of the Twin-Candidate Model

The basic idea of the twin-candidate model is to learn a binary classifier which could judge the preference between candidates of an anaphor. In this section we will describe a general framework of such a model.

### 2.1 Instance Representation

In the twin-candidate model, an instance takes a form like  $i\{C_1, C_2, M\}$ , where  $M$  is a possible anaphor and  $C_1$  and  $C_2$  are two of its antecedent candidates. We stipulate that  $C_2$  should be closer to  $M$  than  $C_1$  in distance. An instance is labelled as “10” if  $C_1$  is preferred to  $C_2$  to be the antecedent, or “01” if otherwise.

A feature vector would be specified for an instance. The features may describe the lexical, syntactic, semantic and positional relationships between  $M$  and each one of the candidates,  $C_1$  or  $C_2$ . In addition, inter-candidate features could be used to represent the relationships between the pair of candidates, e.g. the distance between  $C_1$  and  $C_2$  in position.

### 2.2 Training Procedure

For each anaphor  $M_{ana}$  in a given training text, its closet antecedent,  $C_{ante}$ , would be selected as the anchor candidate to compare with other candidates.

A set of “10” instances,  $i\{C_{ante}, C_p, M_{ana}\}$ , is generated by pairing  $M_{ana}$  and  $C_{ante}$ , as well as each of the intervening candidates  $C_p$ . Also a set of “01” instances,  $i\{C_a, C_{ante}, M_{ana}\}$ , is created by pairing  $C_{ante}$  and each non-antecedental candidate  $C_a$  before  $C_{ante}$ .

**Table 1.** An example text

*[<sub>1</sub> Globalstar] still needs to raise [<sub>2</sub> \$600 million], and [<sub>3</sub> Schwartz] said [<sub>4</sub> that company] would try to raise [<sub>5</sub> the money] in [<sub>6</sub> the debt market] .*

Consider the example in Table 1. In the text segment, [<sub>4</sub> that company] and [<sub>5</sub> the money] are two anaphors with [<sub>1</sub> Globalstar] and [<sub>2</sub> \$600 million] being their antecedents respectively. Thus the training instances to be created for this text would be:

$i\{[1 Globalstar], [2 $600 million], [4 that company]\}$	: 10
$i\{[1 Globalstar], [3 Schwartz], [4 that company]\}$	: 10
$i\{[1 Globalstar], [2 $600 million], [5 the money]\}$	: 01
$i\{[2 $600 million], [3 Schwartz], [5 the money]\}$	: 10
$i\{[2 $600 million], [4 that company], [5 the money]\}$	: 10

Based on the training instances, a classifier is trained using a certain machine learning algorithm. Given the feature vector of a test instance, the classifier would return “10” or “01” indicating which one of the two candidates under consideration is preferred.

### 2.3 Resolution

After the classifier is ready, it could be employed to select the antecedent for an encountered anaphor. The resolution algorithm is shown in Figure 1. In the algorithm, a round-robin model is employed, in which each candidate is compared with every other candidate and the final winner is determined by the won-lost records. The round-robin model would be fair for each competitor and the result is reliable to represent the rank of the candidates.

As described in the algorithm, after each match between two candidates, the record of the winning candidate (i.e., the one judged as preferred by the classifier) will increase and that of the loser will decrease. The algorithm simply uses a unit of one as the increment and decrement. Therefore, the final record of a candidate is its won-lost difference in the round-robin matches. Alternatively, we can use the confidence value returned by the classifier as the in(de)crement, while we found no much performance difference between these two recording strategies in experiments.

---

```

algorithm ANTE-SEL
input:
M: the anaphor to be resolved
candidate_set: the set of antecedent candidates of M,
{C1, C2, ..., Ck}

for i = 1 to K
  Score[ i ] = 0;
for j = K downto 2
  for i = j - 1 downto 1
    /*CR returns the classification result*/
    if CR(i{Ci, Cj, M}) ) = = 10 then
      Score[ i ]++;
      Score[ j ]--;
    if CR(i{Ci, Cj, M}) ) = = 01 then
      Score[ i ]--;
      Score[ j ]++;
  SelectedIdx = arg maxi ∈ candidate_set Score[i];
return CSelectedIdx

```

**Fig. 1.** The original antecedent selection algorithm

### 3 Modified Framework for Coreference Resolution Task

#### 3.1 Non-anaphor Processing

In the task of coreference resolution, it is often that an encountered NP is non-anaphoric, that is, no antecedent exists among its possible candidates. However, the resolution algorithm described in the previous section would always try to pick out a “best” candidate as the antecedent for each given NP, and thus could not be applied for coreference resolution directly.

One natural solution to this is to use an anaphoricity determination (AD) module to identify the non-anaphoric NPs in advance (e.g. [5]). If an NP is judged as anaphoric, then we deploy the resolution algorithm to find its antecedent. Otherwise we just leave the NP unresolved. This solution, however, would heavily rely on the performance of the AD module. Unfortunately, the accuracy that most state-of-the-art AD systems could provide is still not high enough (around 80% as reported in [7]) for our coreference resolution task.

Another possible solution is to set a threshold to avoid selecting a candidate that wins with low confidence (e.g. [6]). Specifically, for two candidates in a match, we update their match records only if the confidence returned from the classifier is above the specified threshold. If no candidate has a positive record in the end, we deem the NP in question as non-anaphoric and leave it unresolved. In other words, a NP would be resolved to a candidate only if the candidate won at least one competitor with confidence above the threshold.

The assumption under this solution is that the classifier would return low confidence for the test instances formed by non-anaphors. Although it may be true,

there exist other cases for which the classifier would also assign low confidence values, for example, when the two candidates of an anaphoric NP both have strong or weak preference. The solution of using threshold could not discriminate these different cases and thus may not be reliable for coreference resolution.

In fact, the above problem could be addressed if we could teach the classifier to explicitly identify the cases of non-anaphors, instead of using threshold implicitly. To do this, we need to provide a special set of instances formed by the non-anaphors to train the classifier. Given a test instance formed by a non-anaphor, the newly learned classifier is supposed to give a class label different from the instances formed by anaphors. This special label would indicate that the current NP is a non-anaphor, and no preference relationship is held between the two candidates under consideration. In this way, the twin-candidate model could do the anaphoricity determination by itself, without any additional pre-processing module. We will describe the modified training and resolution procedures in the subsequent subsections.

### 3.2 Training

In the modified learning framework, an instance also takes a form like  $i\{C_1, C_2, M\}$ . During training, for an encountered anaphor, we create “01” or “10” training instances in the same way as in the original learning framework, while for a non-anaphor  $M_{non\_ana}$ , we

- From the candidate set, randomly select a candidate  $C_{rand}$  as the anchor candidate.
- Create an instance by pairing  $M_{non\_ana}$ ,  $C_{rand}$ , and each of the candidates other than  $C_{rand}$ .

The above instances formed by non-anaphors would be labelled as “00”. Note that an instance may have a form like  $i\{C_a, C_{rand}, M_{non\_ana}\}$  if candidate  $C_a$  is preceding  $C_{rand}$ , or like  $i\{C_{rand}, C_p, M_{non\_ana}\}$  if candidate  $C_p$  is following  $C_{rand}$ .

Consider the text in Table 1 again. For the non-anaphors [3 Schwartz] and [6 the debt market], supposing the selected anchor candidates are [1 Globalstar] and [2 \$600 million], respectively. The “00” instances generated for the text are:

$i\{[1 \text{ Globalstar}], [2 \text{ \$600 million}], [3 \text{ Schwartz}]\}$	: 00
$i\{[1 \text{ Globalstar}], [2 \text{ \$600 million}], [6 \text{ the debt market}]\}$	: 00
$i\{[2 \text{ \$600 million}], [3 \text{ Schwartz}], [6 \text{ the debt market}]\}$	: 00
$i\{[2 \text{ \$600 million}], [4 \text{ that company}], [6 \text{ the debt market}]\}$	: 00
$i\{[2 \text{ \$600 million}], [5 \text{ the money}], [6 \text{ the debt market}]\}$	: 00

### 3.3 Resolution

The “00” training instances are used together with the “01” and “10” ones to train a classifier. The resolution procedure is described in Figure 2. Like in the original algorithm, each candidate is compared with every other candidate. The

difference is that, if two candidates are judged as “00” in a match, both candidates would receive a penalty of  $-1$  in their respective record; If no candidate has a positive final score, then the NP would be deemed as non-anaphoric and left unresolved. Otherwise, it would be resolved to the candidate with highest score as usual. In the case when an NP has only one antecedent candidate, a pseudo-instance is created by paring the candidate with itself. The NP would be resolved to the candidate if the return label is not “00”.

Note that in the algorithm a threshold could still be used, for example, to update the match record only if the classification confidence is high enough.

---

```

algorithm ANTE-SEL
input:
M: the new NP to be resolved
candidate_set: the candidates set of M, {C1, C2, ..., Ck}
for i = 1 to K
    Score[ i ] = 0;
for j = K downto 2
    for i = j - 1 downto 1
        if CR(i{Ci, Cj, M}) ) = = 10 then
            Score[ i ]++;
            Score[ j ]--;
        if CR(i{Ci, Cj, M}) ) = = 01 then
            Score[ i ]--;
            Score[ j ]++;
        if CR(i{Ci, Cj, M}) ) = = 00 then
            Score[ i ]--;
            Score[ j ]--;

    SelectedIdx = arg maxi Ci ∈ candidate_set Score[i];
    if (Score[SelectedIdx] <= 0)
        return nil;
    return CSelectedIdx;

```

**Fig. 2.** The new antecedent selection algorithm

## 4 Evaluation and Discussion

### 4.1 Experiment Setup

The experiments were done on the newswire domain, using MUC coreference data set (Wall Street Journal articles). For MUC-6 [8] and MUC-7 [9], 30 “dry-run” documents were used for training as well as 20-30 documents for testing. In addition, another 100 annotated documents from MUC-6 corpus were also prepared for the purpose of deeper system analysis. Throughout the experiments, C5 was used as the learning algorithm [10]. The recall and precision rates of the coreference resolution systems were calculated based on the scoring scheme proposed by Vilain et al. [11].

**Table 2.** Features for coreference resolution using the twin-candidate model

Features describing the new markable $M$ :	
1. M_DefNP	1 if $M$ is a definite NP; else 0
2. M_IndefNP	1 if $M$ is an indefinite NP; else 0
3. M_ProperNP	1 if $M$ is a proper noun; else 0
4. M_Pronoun	1 if $M$ is a pronoun; else 0
Features describing the candidate, $C_1$ or $C_2$ , of $M$	
5. candi_DefNP_1(2)	1 if $C_1$ ( $C_2$ ) is a definite NP; else 0
6. candi_IndefNP_1(2)	1 if $C_1$ ( $C_2$ ) is an indefinite NP; else 0
7. candi_ProperNP_1(2)	1 if $C_1$ ( $C_2$ ) is a proper noun; else 0
8. candi_Pronoun_1(2)	1 if $C_1$ ( $C_2$ ) is a pronoun; else 0
Features describing the relationships between $C_1$ ( $C_2$ ) and $M$ :	
9. Appositive_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ are in an appositive structure; else 0
10. NameAlias_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ are in an alias of the other; else 0
11. GenderAgree_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ agree in gender; else 0 if disagree; -1 if unknown
12. NumAgree_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ agree in number; else 0 if disagree; -1 if unknown
13. SentDist_1(2)	Distance between $C_1$ ( $C_2$ ) in sentences
14. HeadStrMatch_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ match in head string; else 0
15. NPStrMatch_1(2)	1 if $C_1$ ( $C_2$ ) and $M$ match in full strings; else 0
16. StrSim_1(2)	The ratio of the common strings between $C_1$ ( $C_2$ ) and $M$ , over the strings of $C_1$ ( $C_2$ )
17. SemSim_1(2)	The semantic agreement of $C_1$ ( $C_2$ ) against $M$ in WordNet
Features describing the relationships between $C_1$ and $C_2$	
18. inter_SentDist	Distance between $C_1$ and $C_2$ in sentences
19. inter_StrSim	0, 1, 2 if $StrSim_1(C_1, M)$ is equal to, larger or less than $StrSim_1(C_2, M)$
20. inter_SemSim	0, 1, 2 if $SemSim_1(C_1, M)$ is equal to, larger or less than $SemSim_1(C_2, M)$

The candidates of a markable to be resolved were selected as follows. During training, for each encountered markable, the preceding markables in the current and previous four sentences were taken as the candidates. During resolution, for a non-pronoun, all the preceding markables were included into the candidate set, while for a pronoun, only the markables in the previous four sentences were used, as the antecedent of a pronoun usually occurs in a short distance.

For MUC-6 and MUC-7, our modified framework generated 207k training instances, three times larger than the single-candidate based system by Soon et al [3]. Among them, the ratio of ‘00’, ‘01’ and ‘10’ instances was around 8:2:1. The distribution of the class labels was more balanced than in Soon et al.’s system, where only 5% training instances were positive while others were all negative.

In our study we only considered domain-independent features that could be obtained with low computational cost but with high reliability. Table 2 summarizes the features with their respective possible values. Features  $f_1$ - $f_{17}$  record

the properties of a new markable and its two candidates, as well as their relationships. Most of these features could be found in previous systems on coreference resolution (e.g. [3], [4]). In addition, three inter-candidate features, *f18-f20*, mark the relationship between the two candidates. The first one, *inter\_SentDist*, records the distance between the two candidates in sentences, while the latter two, *inter\_StrSim* and *inter\_SemSim* compare the similarity scores of the two candidates, in string-matching and semantics respectively.

To provide necessary information of feature computation, an input raw text was preprocessed automatically by a pipeline of NLP components. Among them, the chunking component was trained and tested for the shared task for CoNLL-2000 and achieved 92% F-score. The HMM based NE recognition component was capable of recognizing the MUC-style NEs with F-scores of 96.9% (MUC-6) and 94.3% (MUC-7).

## 4.2 Results and Discussion

In the experiment we compared four systems:

**SC.** The system based on the single-candidate model. It was a duplicate of the system by Soon et al. [3]. The feature set used in the baseline system was similar to those listed in Table 2, except that no inter-candidate feature would be used and only one set of features related to the single candidate was required.

**TC\_AD.** The system based on the twin-candidate mode with the original learning framework, in which non-anaphors were eliminated by an anaphoricity determination module in advance. We built a supervised learning based AD module similar to the system proposed by Ng and Cardie [7]. We trained the AD classifier on the additional 100 MUC-6 documents. By adjusting the misclassification cost parameter of C5, we obtained a set of classifiers capable of identifying “positive” anaphors with variant recall and precision rates.

**TC\_THRESH.** The system based on the twin-candidate mode with the original learning framework, using threshold to discard the low-confidenced comparison results between candidates.

**TC\_NEW.** The system based on the twin-candidate mode, with our modified learning framework.

The results of the four systems on MUC-6 and MUC-7 are summarized in Table 3. In these experiments, five-fold cross-evaluation was performed on the training data to select the resolution parameters, for example, the threshold for systems TC\_THRESH and TC\_NEW, and final AD classifier for TC\_AD.

As shown in the table, the baseline system SC achieves 66.1% and 65.9% F-measure for MUC-6 and MUC-7 data sets. This performance is better than that reported by Soon et al. [3], and is comparable to that of the state-of-the-art systems on the same data sets.

From the table we could find system TC\_AD achieves a comparatively high precision but a low recall, resulting in a F-measure worse than that of SC. The analysis

**Table 3.** The performance of different coreference resolution systems

Experiments	30 Docs						100 Docs					
	MUC-6			MUC-7			MUC-6			MUC-7		
	R	P	F	R	P	F	R	P	F	R	P	F
SC	70.4	62.4	66.1	69.8	62.5	65.9	67.9	62.1	64.9	69.8	62.5	65.9
TC_AD	62.6	66.4	64.4	60.8	64.7	62.7	61.6	65.4	63.4	60.8	64.6	62.7
TC_THRESH	<b>70.7</b>	59.1	64.4	<b>70.0</b>	61.7	65.6	<b>71.0</b>	60.7	65.4	<b>70.6</b>	60.9	65.4
TC_NEW	64.8	<b>70.1</b>	<b>67.3</b>	66.0	<b>68.6</b>	<b>67.2</b>	67.0	<b>70.2</b>	<b>68.5</b>	67.0	<b>69.2</b>	<b>68.1</b>

of the AD classifier reveals that it successfully identifies 79.3% anaphors (79.48% precision) for MUC-6, and 70.9% anaphors (76.3% precision) for MUC-6. That means, although the pre-processing AD module could partly avoid the wrong resolution of a non-anaphor, it eliminates many anaphors at the same, which leads to the low recall for coreference resolution. Although in resolution different AD classifiers could be applied, we only observe the tradeoff between recall and precision, with no effective resolution improvement in F-measure.

In contrast to TC\_AD, system TC\_THRESH yields large gains in recall. The recall, up to above 70%, is higher than all the other three systems. However, the precision at the same time is unfortunately the lowest. Such a pattern of high recall and low precision indicates that using threshold could reduce, to some degree, the risk of eliminating true anaphors, but it would be too lenient to effectively block the resolution of non-anaphors.

Compared with TC\_AD and TC\_THRESH, TC\_NEW produces large gains in the precision rates, which rank the highest among all the four systems. Although the recall also drops at the same time, the increase in the precision could compensate it well; we observe a F-measure of 67.3% for MUC-6 and 67.2% for MUC-7, significantly better ( $p \leq 0.05$ , by a sign test) than the other twin-candidate based systems. These results suggest that with our modified framework, the twin-candidate model could effectively identify non-anaphors and block their invalid resolution, without affecting the accuracy of the antecedent determination for anaphors.

In our experiment we were interested to evaluate the resolution performance of TC\_NEW under different sizes of training data. For this purpose, we used the additional 100 annotated documents for training, and plotted the learning curve in Figure 3. The curve indicates that the system could perform well with a small number of training data, while the performance would get further improved with more training data (the best performance is obtained on 90 documents).

In Table 3, we also summarized the results of different systems trained on 100 documents. In contrast to TC\_NEW, we find for system SC, there is no much performance difference between using 30 and 100 training documents. This is consistent with the report by Soon et al. [3] that the single-candidate model would achieve the peak performance with a moderate size of data. In the table we could also find that the performance improvement of TC\_NEW against the other three systems is apparently larger on 100 training documents than on 30 documents.

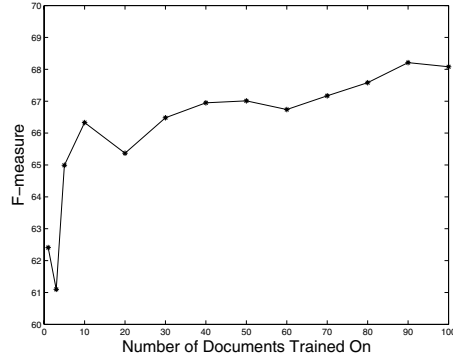


Fig. 3. Learning curve of system TC\_NEW on MUC-7

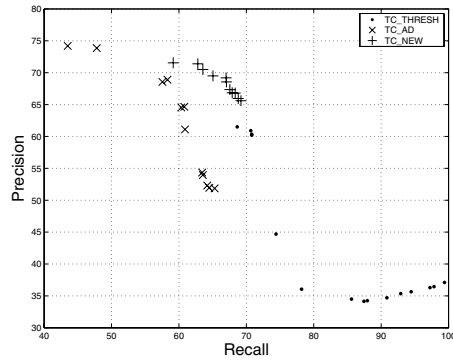
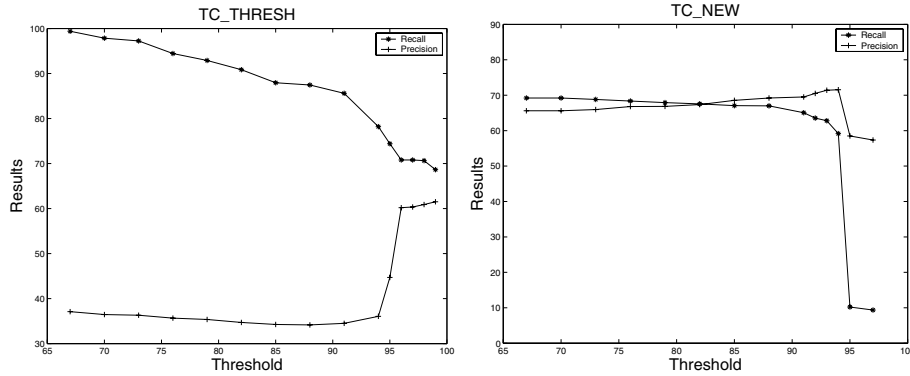


Fig. 4. Recall and precision results for the twin-candidate based systems

In Figure 4, we plotted the variant recall and precision scores that the three twin-candidate based systems were capable of producing when trained on 100 documents. (Here we only showed the results for MUC-7. Similar results could be obtained for MUC-6). In line with the results in Table 3, system TC\_AD tends to obtain a high precision but low recall, while system TC\_THRESH tends to obtain a high recall but low precision. Comparatively, system TC\_NEW produces even recall and precision. For the range of recall within which the three systems coincide, TC\_NEW yields higher precision than the other two systems. This figure further proves the effectiveness of our modified learning framework.

As mentioned, in systems TC\_THRESH and TC\_NEW the threshold parameter could be adjusted. It would be interesting to evaluate the influence of different thresholds on the resolution performance. In Figure 5 we compared the recall and precision of two systems, with thresholds ranging from 65 to 100.

In TC\_THRESH, when the threshold is low, the recall is almost 100% while the precision is quite low. In such a case, all the markables, regardless anaphors or



**Fig. 5.** Performance of TC\_THRESH and TC\_NEW under different thresholds

non-anaphors, will be resolved. As a consequence, all the occurring markables in a document tends to be linked together. In fact, the effective range of the threshold that leads to an acceptable performance is quite short. The threshold would only work when it is considerably high (above 95). Before that, the precision remains very low (less than 40%) while the recall keeps going down with the increase of the threshold.

By contrast, in TC\_NEW, both the recall and precision vary little unless the threshold is extremely high. That means, the threshold would not impose much influence on the resolution performance of TC\_NEW. This should be because in the modified framework, the cases of non-anaphors are determined by the special class label “00”, instead of the threshold as in TC\_THRESH. The purpose of using threshold in TC\_NEW is not to identify the non-anaphors, but to improve the accuracy of class labelling. Indeed, we could obtain a good result without using any threshold in TC\_NEW. These further confirm our claims that the modified learning framework could perform more reliably than the solution of using threshold.

## 5 Conclusions

In this paper we aimed to find an effective way to apply the twin-candidate model into coreference resolution task. We proposed a modified learning framework in which non-anaphors were utilized to create a special class of training instances. With such instances, the resulting classifier could avoid the invalid resolution of non-anaphors, which enables the twin-candidate model to be directly deployed to coreference resolution, without using an additional anaphoricity determination module or using a pre-defined threshold.

In the paper we evaluated the effectiveness of our modified framework on the MUC data set. The results show that the system with the new framework outperforms the single-candidate based system, as well as the twin-candidate

based systems using other solutions. Especially, the analysis of the results indicates that our modified framework could lead to more reliable performance than the solution of using threshold. All these suggest that the twin-candidate model with the new framework is effective for coreference resolution.

## References

1. McCarthy, J., Lehnert, Q.: Using decision trees for coreference resolution. In: Proceedings of the 14th International Conference on Artificial Intelligences. (1995) 1050–1055
2. Connolly, D., Burger, J., Day, D. New Methods in Language Processing. In: A machine learning approach to anaphoric reference. (1997) 133–144
3. Soon, W., Ng, H., Lim, D.: A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics* **27** (2001) 521–544
4. Ng, V., Cardie, C.: Improving machine learning approaches to coreference resolution. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia (2002) 104–111
5. Yang, X., Zhou, G., Su, J., Tan, C.: Coreference resolution using competition learning approach. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, Japan (2003)
6. Iida, R., Inui, K., Takamura, H., Matsumoto, Y.: Incorporating contextual cues in trainable models for coreference resolution. In: Proceedings of the 10th Conference of EACL, Workshop “The Computational Treatment of Anaphora”. (2003)
7. Ng, V., Cardie, C.: Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In: Proceedings of the 19th International Conference on Computational Linguistics (COLING02). (2002)
8. MUC-6: Proceedings of the Sixth Message Understanding Conference. Morgan Kaufmann Publishers, San Francisco, CA (1995)
9. MUC-7: Proceedings of the Seventh Message Understanding Conference. Morgan Kaufmann Publishers, San Francisco, CA (1998)
10. Quinlan, J.R.: C4.5: Programs for machine learning. Morgan Kaufmann Publishers, San Francisco, CA (1993)
11. Vilain, M., Burger, J., Aberdeen, J., Connolly, D., Hirschman, L.: A model-theoretic coreference scoring scheme. In: Proceedings of the Sixth Message understanding Conference (MUC-6), San Francisco, CA, Morgan Kaufmann Publishers (1995) 45–52