Journal of Computational Information Systems 9: 1 (2013) 1–8
Available at http://www.Jofcis.com

An Instance based Dynamic Generalization Model for Coreference Resolution

Muyu Zhang [1], Qin Bing [1,*], Liu Ting [1]

[1] *Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology, Harbin 150001, China*

Abstract

The traditional coreference resolution methods use immutable models to handle all kinds of instances, once the model created, all the parameters and weight are fixed, and thus acquire good results in common examples, but neglect long tail instances. To deal with this problem, we present an instance based dynamic generalization algorithm (IDGen). IDGen constructs an exclusive model level by level automatically for every single instance with the generation points created from the features. In every generalization level, we choose the best generalization point according to some criterions. The experiment shows that our IDGen algorithm is superior to the traditional methods.

Keywords: Coreference Resolution; Dynamic Generalization; Generalization Point; IDGen

1 Introduction

Coreference resolution is the process of identifying mentions referring to the same real world entity or concept. As a key problem in discourse and language understanding, coreference resolution is crucial in many natural language applications, such as question answering, text summarization, information retrieval, machine translation, information extraction and so on.

Previous coreference resolution approaches usually based on heuristic knowledge or statistical learners. The heuristic approaches focus on linguistics knowledge such as syntactic constraints [1], in particular centering algorithms [2] that pay attention to the center transitions to forecast next focus which can be used in coreference resolution.

The focus of coreference resolution underwent a shift to machine learning approaches [3] partly attributed to the advent of the statistical NLP era. Until recently, most approaches tried to solve the problem by binary classification, where the probability of a pair of mentions being coreferent is estimated from labeled data using local context information. Yang presents a twin-candidate model to extend traditional mention pair of "anaphora, antecedent" to "anaphora, coreferencial antecedent, uncorefenrencial antecedent", so as to learn the differences between positive and negative instances [14]. Nicolae proposes a graph cutting algorithm for consistency assurance [4]. Yatskevich describes a cross document method which uses the RDF graph of entities to compute the similarity, so as to merge the entities from different documents accurately [5]. Demis uses integer linear programming to decrease the collisions during the process of merging [6]. Ren presents a combination of machine learning

*Corresponding author.
Email address: myzhang@ir.hit.edu.cn (Qin Bing).

methods for more effective feature combination [7]. Nemulapalli uses Bagging method for Classifier combination to get better result [8]. Zhou combines tree kernel and dependence analysis for anaphora resolution [9]. Rahman incorporates world knowledge to improve the performance [10].

Besides binary classification, clustering method is also used in coreference resolution. Wangatff proposes a noun phrase resolution system based on clustering algorithms in which a feature vector will be extracted for every noun phrase. With the vector, the model decides which class the phrase belongs to base on the distance threshold [11]. Ng uses N best clustering method to evaluate the number of classes [12]. Poon presents a model based on the Markov logic to estimate the number of clusters [13].

The previous coreference resolution approaches usually cover common examples, but neglect the long tail instances because they use invariable models and fixed parameters to handle all kinds of mentions, but neglect the differences between instances. To handle this problem, we present the instance based dynamic generalization algorithm (IDGen). In our model, all the training instances will be considered to make use of the infrequent examples and capture the differences between cases. The IDGen algorithm should build the instance library first to store training instances labeled the tag of class. For every test instance, the IDGen algorithm processes according to the following steps:

(1) Extract all the generalization points
(2) Select the best generalization point according our criterions
(3) Retrieve the instance library to get all training examples with the same value at the certain generalization point. All selected instances construct the new set “reference set A”.
(4) Whether to terminate: Yes, go to step (5); No, repeat step (2)(3) based on the new reference set ‘A’.
(5) Choose the final class tag, finished.

In our method, instead of invariable model, a special generalization model will be constructed for every test instance which suits the special case best. The algorithm based on instance matching can achieve better results in the infrequent instances and get an improvement on the long tail examples. Our experiments achieve better results than the traditional method.

2 Approach Overview

The main idea behind the IDGen algorithm is that instances with similar features should have the same class tag. In our method these features will be converted into generation points. The algorithm uses the generation points of test case to retrieve the instances library for certain times and select all the similar instances with the same value. The final class of test instance will be decided based on the category distribution of the final reference set constructed by similar instances.

2.1 Dynamic generalization algorithm

Figure 1 shows the illustration of the process. The instance library is constructed of training instances whose generalization points have been extracted. The IDGen algorithm indexes all the generalization points for further retrieval. Exactly, the model will execute according to the following steps:

(1) For every training instance, extract all the generation points to construct the instances library first.
(2) For a test instance, extract all generalization points and set the instance library to be the reference set.
(3) Choose the best generalization point from the unused ones according to some criterions, then retrieve the reference set to select all the instances with the same value to the test instance at the best point. All the training instances retrieved from the instance library construct the new reference set.
(4) Check the new reference set: If all the training instances belong to the same class or all generalization points have been used, algorithm terminates to choose the final class for the test instance according to the category distribution within the final reference set. Otherwise, back to step (3).

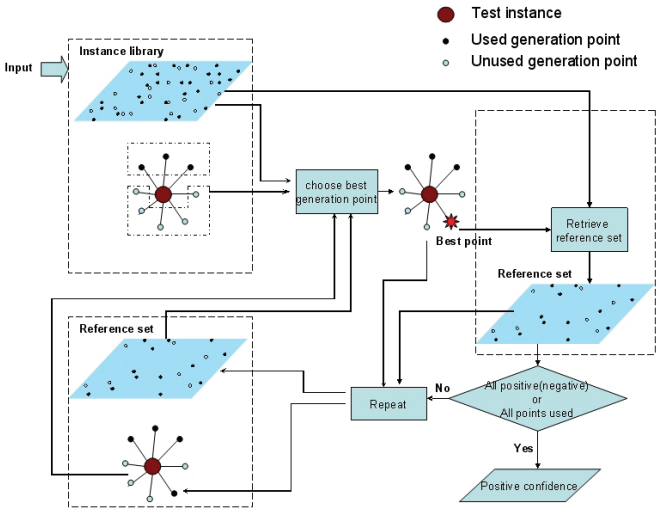

Fig. 1: Description of $IDGen$ mechanism (Section2.1). Construct the instance library first, select the best generalization point. Retrieve the reference set using the best generalization point value to construct a new reference set. Repeats until the set satisfies a certain condition to decide the final class.

In the description of step (1), the way to extract all the generalization points is the basic of the IDGen algorithm model. In step (3), how to choose the best generalization points is another key point. In step (4), the method to compute the positive confidence based on the reference set is very important too. This paper will list all the detailed explanation for these problems in Chapter 3. In the next paragraph the process of retrieving the reference set will be presented using the best generalization point. There are two kinds of generalization: level generalization and vertical generalization.

2.2 Level and vertical generalization

In our method, the classification of test instance has been transformed into the process of selecting the best generalization points and retrieving the reference set. There are two kinds of generalization: level and vertical generalization.To classify a test instance, the algorithm needs both level and vertical generalization. They work together to choose the best class label.

Level generalization: In the certain retrieval of reference set, the IDGen method should select the best one from all the unused generalization points. See Figure.2: to deal with a test case, the algorithm should choose the first best generalization point and use the best point to retrieve the reference set. This is the first level generalization and then the second and third one. Selecting the best points is the process of level generalization, which may happen several times during processing one test instance.

Vertical generalization: The IDGen algorithm may retrieve the reference set step by step for many times and all the level generalizations form the process of vertical generalization. See Figure.2: To deal with a test instance, the algorithm may finish the first level generalization with the whole instance library as the reference set and check whether the new set after retrieval can meet the requirement of our criterions; if not, the second level generalization start. All the level generalization will affect each other and work together to construct the vertical generalization.

3 Key Point

3.1 Generalization point extraction

The key points of IDGen algorithm are illustrated in Chapter 2. Now we will give the detailed explanation.

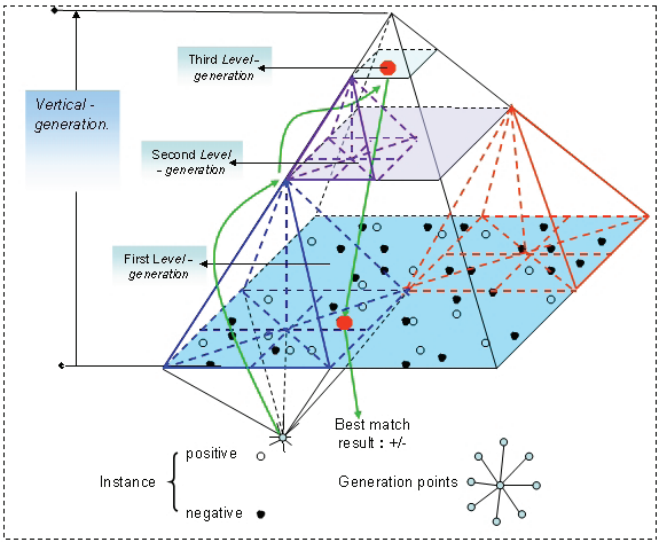

Fig. 2: Two kinds of generalization (Section2.2). For every test instance, $IDGen$ extracts all the generalization points first and chooses the first best one at the first level, get a new reference set. Repeat until the condition satisfies to decide the final class.

In the previous description, we use the features to depict the similar instances. In this paper we convert these features into generalization points.For every instance, both training and testing. To deal with a document: the IDGen algorithm must preprocess it first and create all the instance pairs. Then the algorithm extracts all the features such as "Mention type of antecedent"、"Gender"; Finally, the algorithm converts all the features into generalization points, such as "A mention type: NAM"、"AB gender: True", and so on.

To describe all the generalization points clearly, we use A to denote antecedent; B to denote anaphora; AB to denote the union of antecedent and anaphora. For instance ¡A/B¿, its generalization points usually have the following form: [A/B/AB][Feature name][Feature value].

3.2 Select best generalization point

Criterion: the goal of selecting generalization points is to retrieve the best instances similar with test example so as to supply more decision information. Now we have tried five methods below.

Min_Info_Entropy: This criterion consults the information gain of ID3 algorithm based on the theory of information entropy. Specifically, the set S consists of two kinds of instances: positive instances (p), negative instances (n). The information entropy of set S is computed as below:

$$I(S) = -\frac{p}{p+n}\log 2\frac{p}{p+n} - \frac{n}{n+p}\log 2\frac{n}{n+p} \tag{1}$$

$I(S)$ is the average amount of information needed to classify a instance correctly. The IDGen algorithm is different from decision tree because the generalization point is selected during the testing process of retrieving reference set. The value of feature is certain when the algorithm choose the best generalization point so that the information gain should be computed as below:

$$Gain(A_0) = I - I(A_0) \tag{2}$$

$$A_0{}^* = \max_{A_0} Gain(A_0) = \max_{A_0}\{I - I(A_0)\} = \min_{A_0} I(A_0) \tag{3}$$

A_0 is a feature of test instance whose value is certain, so it's equal to the feature value. In fact, A_0 is the best generalization point we want. See Formula 3, $I(A_0)$ means the information entropy of the instance set selected by

the A_0. This criterion prefers the generalization point which can retrieve the instance set with the least information entropy. The Formula 3 can be converted into the form below:

$$A_0{}^* = \min_{A_0} I(A_0) = \max_{A_0} |\frac{p_0}{p_0 + n_0} - \frac{n_0}{n_0 + p_0}| \tag{4}$$

Min_Info_Entropy focuses on the distinguishöf the final reference set. Less information entropy means less balance in data and result in a positive confidence near to 0 or 1, so the IDGen algorithm can choose the correct class more easily.

Max_Inst_Amount: The Min_Info_Entropy criterion based on information entropy is usually sensitive to the quality of data, but training data usually contains pollution, especially during the processing and feature extraction. In some extreme cases, there is only one instance in the reference set after retrieval by certain generalization point, once polluted in the generalization point, our model will make a mistake.

To handle this problem, we present the Max_Inst_Amount based on the instances number. This criterion prefers the generalization point whose reference set contains the most instances, so as to enrich the cases as evidence. Max_Inst_Amount can reduce the affection of pollution effectively, but it neglects the conflict between evidence. There are both large number of positive instances and negative instances in the reference set, but the amounts of them are similar and this will lead to a low confidence in category prediction.

Min_Inst_Amount: Max_Inst_Amount focuses on the number of evidence and tries to select the generalization point who gets the reference set with the largest number of instances. But the more instances, the more time it will cost. So we present the Min_Inst_Amount criterion to minimize the reference set so as to accelerate the testing process.

Max_PosInst_Amount: The three criterions before have no bias in category, but most of instances in coreference resolution is negative and the data set is unbalanced. For example, using the method of Soon to construct the training instances from ACE2005 English corpora, the ratio between the number of negative and positive examples is 1:19.

The unbalance of training data will make the model incline to choose the category with more instances. In the task of coreference resolution, the model will incline to label the test case negative, it will decrease the recall. So we present the Max_PosInst_Amount criterion to focus on the number of positive instances.

Max_PosInst_Ratio: Similar with last criterion, the Max_PosInst_Ratio is designed to deal with the unbalance of data. Different from Max_PosInst_Amount, this criterion focuses on the proportion of positive instances in the reference set, so as to give direct great impact on the positive confidence.

3.3 Compute positive confidence

The criterions above are designed to select the most similar instances from the training instance library. To decide when to stop the process of dynamic generalization, the IDGen algorithm computes the positive confidence based on the category distribution within the final reference set.

The common approach to compute the positive confidence is to set the concept as the proportion of positive instance. If our algorithm adopts this method, the model will classify the test instance as a positive one when the proportion of positive examples exceeds 50%; otherwise, the test instance will be negative. One assumption behind the method is that the weight of positive instances is equal to the negative instances. But the data of coreference resolution is unbalanced, so positive instances are more important than negative instances, especially when we lack of feature information. So we use Formula 5 to show the importance of positive instance:

$$Confidence(p) = \begin{cases} \frac{p}{2r} & \text{when } 0 \leq x \leq r, \\ \frac{p-r}{2(1-r)} + 0.5 & \text{when } r < x \leq 1. \end{cases} \tag{5}$$

The parameter of p is the proportion of positive instance, r is experimental parameter.

4 Experiments

In this paper we used the ACE2005 English training corpora and experimental setup to evaluate our IDGen algorithm and compare it with the traditional machine learning methods.

4.1 Result on the ACE data

In the experiment we used the ACE2005 English corpora, which consists of BN(226 document) and NW(106 document) data sets. The algorithm preprocesses the corpus first and then constructed instances based on the mentions obtained by preprocessing using the Soon method to construct an end to end system without golden mentions. We use MUC6 approach to evaluate our method during the experiment.The results are presented in Table 1 and 2. We use 5 fold experiments and compare the IDGen method with the traditional machine learning methods.

Table 1: Experimental results in ACE2005 English BN data set. Our method exceeds all the traditional machine learning approaches. In our IDGen algorithm experiments, we use five retrieval standards to find the best one.

Method	Retrieval standard	P	R	F
Maxent	–	0.586	0.524	0.553
J48	–	0.574 0	0.532	0.552
SVM	–	0.570	0.528	0.548
IDGen	Max_Inst_Amount	0.585	0.528	0.548
IDGen	32-Max_PosInst_Ratio	0.589	0.526	0.556
IDGen	64-Min_Info_Entropy	0.586	0.528	0.555
IDGen	32-Min_Inst_Amount	0.627	0.510	0.562
IDGen	64-Max_PosInst_Amount	0.625	0.510	0.561

Table 2: Experiment results in ACE2005 English NW data set. Our method gets the better or equal results than all the traditional machine learning approaches.

Method	Retrieval standard	P	R	F
Maxent	–	0.532	0.617	0.571
J48	–	0.533 0	0.630	0.577
SVM	–	0.527	0.624	0.571
IDGen	Max_Inst_Amount	0.537	0.623	0.576
IDGen	32-Max_PosInst_Ratio	0.538	0.614	0.573
IDGen	64-Min_Info_Entropy	0.537	0.624	0.577
IDGen	32-Min_Inst_Amount	0.538	0.618	0.575
IDGen	64-Max_PosInst_Amount	0.537	0.621	0.575

4.2 Analysis

After analyzing the experiments we find two kinds of affects from our method compared to the traditional machine learning approaches:

(1) The errors made by traditional approaches are corrected by our algorithm. This is positive effect.

(2) The traditional methods make right decisions, but our system makes mistakes. This is negative effect.

The IDGen algorithm needs to reserve the first affect and try to enlarge the positive factor, but the second effect needs to be avoided. After analysis data, we found the reasons of the positive affect. Our approach acquires good results in the common examples and improves the weakness of traditional methods. We achieve lots of improvement in the long tail instances because our instance based algorithm can retrieve all the similar instances in the library no matter how much they are. The decision will be made by the distribution of reference set. So our model can make better choices than the traditional methods if there are some similar instances. But it requires more effective generalization points selecting criterions to retrieve the instance library. There are five criterions used in our method. The most effective criterion is Max_Inst_Amount which has a good robustness, but a little simple. So we need to create new criterion to select generalization points more effective.

But the IDGen method also binges some problems and has some negative effect the experiment. First, to deal with a test instance, we should extract all the generalization points. If one of these generalization points has a special value which will be used to retrieve the reference set, our model may offset by the infrequent value to make mistakes. Second, our criterions used to select generalization points still have limitations in their expressiveness.

Generally speaking, our approach got the result that exceeds the traditional machine learning methods such as J48, SVM, Maxent in ACE2005 English data.After analysis of ACE2005 English data, we found that there are many data noises in the instances for some reasons. First, we use no criterion set of mentions as the candidates. On the contrary, our model will create the candidate mentions by preprocessing the document first. Second, our feature extractors do not have the special domain and import more noises for our instance library.

To summarize, our method can supply more effect feature information and suffer more noises than the traditional methods, especially in the ACE2005 English data sets which lack of feature information and more similar to the real environment. Now our model achieves better performance than the traditional methods.

5 Conclusion

This paper mainly discusses the generalization points selecting criterions and the way to compute the positive confidence. We learn the criterions that used in the process of features selecting from ID3 decision tree algorithm. This paper presents five kinds of criterions used to select generalization points. To handle the unbalance problem in the training instance library, we consider the linear discriminative function of positive instances to define the positive confidence function. The experimental results show that our selecting criterions and positive confidence function improve the experiment results in the ACE2005 English data and get a better result than the traditional machine learning methods.

Our generalization points selecting criterions don't consider enough reference factors and the confidence function is still a little simple. So our experiment results do not satisfy enough now. Therefore, we need to pay more attention to these problems in our future work:

(1) We planned to import more discourse information instead of the greedy manner and combine the count of information with the quality of information to select the best generalization point selecting criterion.

(2) To reduce the affection of data pollution, we planned to make use of the information during the process of generalization point selecting to construct better positive confidence function.

(3) Analyzing the experiment data more carefully and carrying out a more detailed error analysis to find the advantage of our method. Try to combine our method and machine learning approaches and make use of the advantages of them.

Acknowledgement

This work was supported by National Natural Science Foundation of China(NSFC) via grant 61133012, the National 863 Leading Technology Research Project via grant 2012AA011102 and the National Natural Science Foundation of China Surface Project via grant 61073126.

References

[1] J. R. Hobbs. Resolving Pronoun References. Lingua. 1978, 44:311338

[2] B. J. Grosz, S. Weinstein, A. K. Joshi. Centering: A Framework for Modeling the Local Coherence of Discourse. Computational Linguistics. 1995, 21(2):203225

[3] J. McCarthy and W. Lehnert. 1995. Using decision trees for coreference resolution. In: C.R. Perrault ed. Proc. of the Fourteenth International Joint Conference on Artificial Intelligence. Québec, Canada: Springer, 10501055.

[4] C. Nicolae, G. Nicolae. Bestcut: A Graph Algorithm for Coreference Resolution. Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia, 2006. Association for Computational Linguistics, 2006:275-283

[5] M. Yatskevich, C. Welty, J. W. Murdock. Coreference resolution on RDF Graphs generated from Information Extraction: first results. Proc. of ISWC2006 Web Content Mining with Human Language Technology, 2006.

[6] P. Denis, J. Baldridge. Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. Human Language Technologies 2007: The Conference of the North American Chater of the Association for Computational Linguistics, Rochester, New York, 2007. Association for Computational Linguistics, 2007:236243

[7] F. Ren, J. Zhu. An Effective Hybrid Machine Learning Approach for Coreference Resolution. Proceedings of the 6th SIGHAN Workshop on Chinese Language Processing, Hyderabad, India, 2008. 2008:2430

[8] S. Vemulapalli, X. Luo, J. F. Pitrelli，et al. Classifier Combination Techniques Applied to Coreference Resolution. Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium, Boulder, Colorado, 2009. Association for Computational Linguistics, 2009:16

[9] F. Kong, G. Zhou, L. Qian, Q. Zhu. Dependency-driven Anaphoricity Determination for Coreference Resolution. COLING'2010:599-607.

[10] A. Rahman and V. Ng. Coreference resolution with world knowledge. In The 49th Annual Meeting of the Association for Computational Linguistics, 2011, Human Language Technologies, 814-824.

[11] C. Cardie, K. Wagstaf. Noun Phrase Coreference as Clustering. Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, College Park, Maryland, 1999. 1999:8289

[12] V. Ng. Unsupervised Models for Coreference Resolution. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, 2008. Association for Computational Linguistics, 2008:640649

[13] H. Poon, P. Domingos. Joint Unsupervised Coreference Resolution with Markov Logic. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, 2008. Association for Computational Linguistics, 2008:650659

[14] Xiaofeng Yang, Jian Su, and Chew Lim Tan. 2008b. A twin-candidate model for learning-based anaphora resolution. Computational Linguistics, 34(3):327-356.