

NYU's Chinese ACE 2005 EDR System Description

Can Global Re-Ranking and Semantic Role Labeling help Chinese EDR?

Heng Ji

Adam Meyers

Ralph Grishman

Proteus Project

Department of Computer Science

New York University

New York, NY, 10003, USA

(hengji, meyers, grishman)@cs.nyu.edu

Abstract

In this report we present the overall architecture of the NYU Chinese ACE 2005 EDR (Entity Detection and Recognition) system, focussing mainly on mention detection. Central to our research is the idea that the performance of analyzers which come early in the pipeline and use primarily local knowledge, such as name and nominal phrase recognition, can be significantly improved by the more global knowledge of later stages. We use an N-Best approach to generate multiple name hypotheses, and apply parsing, chunking and list matching to generate multiple nominal hypotheses. Then these multiple mention hypotheses are re-scored using the global information of coreference, relations and events, as well as the selectional preferences of semantic roles. In addition, we briefly describe our attempts at re-scoring coreference results with relation and event information, and the use of semi-supervised learning techniques to extend the training data.

1 System Pipeline

The overall architecture of our primary Chinese EDR system is presented in Figure 1. All subtasks are arranged in a sequential model, with the mention detection and coreference stages generating N-Best multiple hypotheses. Then we incorporate information from later stages to re-rank/re-score them. The new top hypothesis is our final extraction result. The shaded components in the figure present the new methods we tried this year. The secondary system keeps the same structure as the primary system, except it does not include the parser, semantic role tagger and semantic role cluster re-scorer (i.e. the dashed components in Figure 1). The primary system required about 8 hours using 5 PC's, primarily for parsing and semantic role tagging; the secondary system required about 80 minutes on a single PC.

2 General Overview of System Components

We built on the base of NYU Chinese ACE04 EDR, which is a relatively simple, cleanly structured system. Except for the re-ranking techniques, we briefly describe each component in the following.

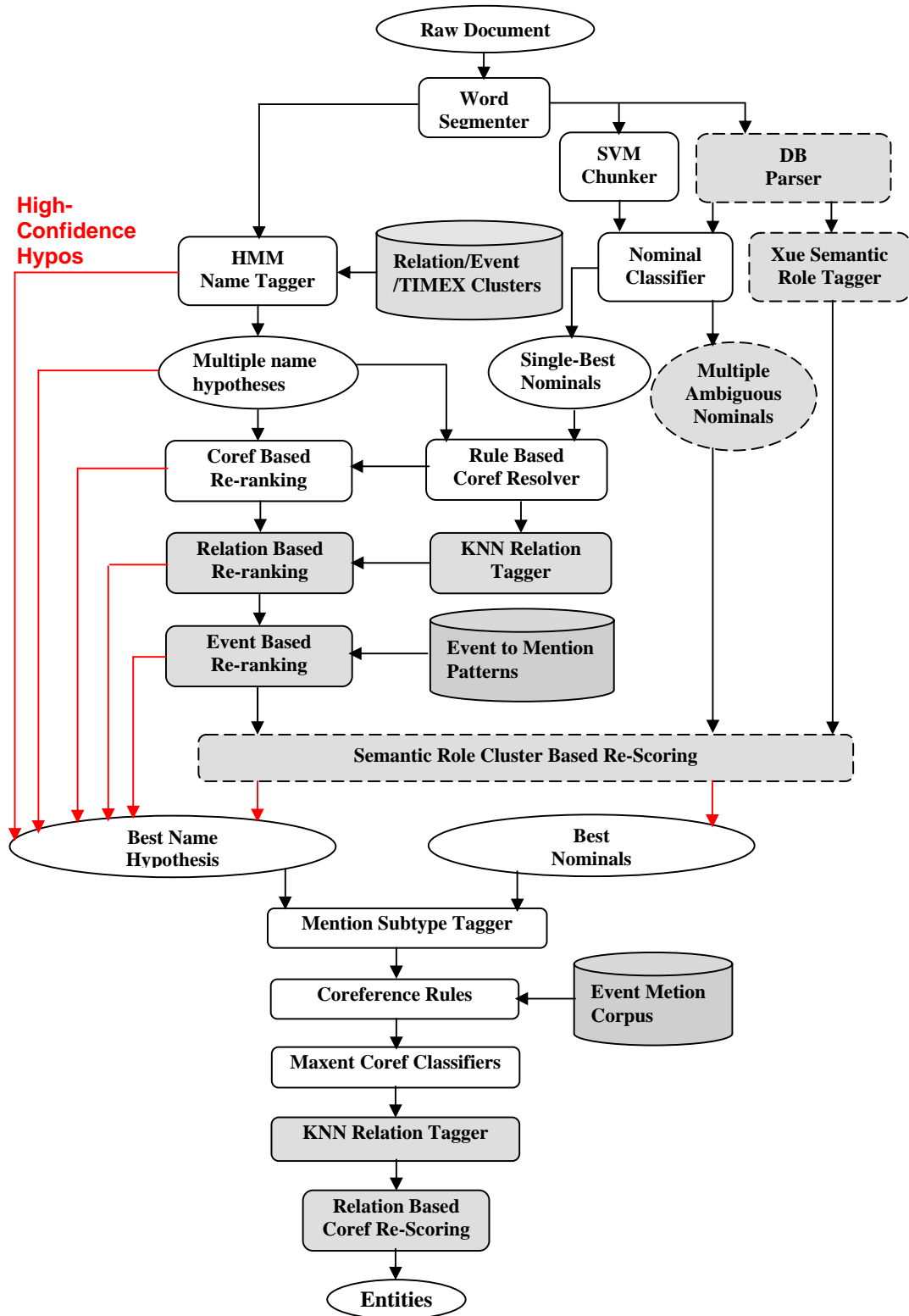


Figure 1. NYU Chinese ACE05 EDR System

2.1 Name Tagger

We apply a HMM tagger to identify the main ACE entity types: Person, GPE, Organization and Location. The HMM tagger generally follows the Nymble model (Bikel et al, 1997), and operates on the output of a word segmenter from Tsinghua University. We have made the following improvements to the model. We extract word clusters from the ACE04 (with types mapped into 05 set) and ACE05 relation training corpus. The set of ACE relations includes several involving employment, social, and family relations. We gathered the words appearing as an argument of one of these relations, eliminated low-frequency terms and manually edited the ten resulting clusters to remove inappropriate terms. These are then combined with lists (of titles, organization name suffixes, location suffixes). If a word is not part of a relation cluster, we consider it an independent (1-word) cluster. The Nymble model relies on a multi-level linear interpolation model for backoff. We extended this model by adding a level from word to cluster, so as to estimate more reliable probabilities for words in these clusters.

To take advantages of Chinese language-specific name structures, we do name structure parsing using an extended HMM which includes a larger number of states (14). This new HMM can handle name prefixes and suffixes, and transliterated foreign names separately. We also extract event trigger word lists and a title list from the ACE05 event training corpus, and extract a TIME word list from TIMEX data, and use them to construct additional features in the Nymble model.

The training data consists of the Beijing Corpus with Location manually separated from GPE, and ACE03, 04, 05 training data, plus the data generated from LDC unannotated data by semi-supervised learning (described in Section 5), in total 4,009,482 words.

The HMM tagger uses the best-first search to generate NBest hypotheses, and also computes the margin – the difference between the log probabilities of the top two hypotheses. This is used as a rough measure of confidence in the top hypothesis. A large margin indicates greater confidence that the first hypothesis is correct. So if the name tagger margin of a sentence is above a threshold, we select the first hypothesis, dropping the others and by-passing later re-ranking/re-scoring steps.

After getting the multiple name hypotheses for each sentence, we then incorporate the results from later components into statistical re-ranking models to determine a new ranking, and output the new top name hypothesis. We then use semantic role clusters to further disambiguate the difficult names. We will describe the details in Section 3 and Section 4.

Finally we apply a set of post-processing heuristic rules to correct some omissions and systematic errors, and identify names in the modifier of a nested organization of the best hypothesis. Furthermore, compared to bnews and nwire, weblog documents include many more new names, particularly names written in English, so we applied some patterns to discover them. For example, in the structure ChineseNAMEA (EnglishNAMEB), if one of A and B is tagged and the other is missed, we identify them by name internal structures, and corefer them to each other.

We apply list matching to identify facility, weapon and vehicle names, due to the small amount of training data.

2.2 Nominal Mention Tagger

We trained both a SVM based chunker and Daniel Bikel’s parser (Bikel 2004) on Chinese Penn TreeBank V5.1 to process the segmented data, and keep the multiple noun phrase boundaries. We classify entity types for the nominal heads by list matching. The lists are generated from ACE 03, 04 and 05 training data, and a small part from Hownet and LDC lists. We keep multiple entity types for high-frequency ambiguous mention heads. We apply context restrictions to identify the nominals in the noun phrase modifiers. Then for those ambiguous mentions, we use a statistical re-scoring model, and also some heuristic patterns/rules based on semantic role clusters (described in Section 4) to select the best set of nominals.

2.3 Entity SubType Tagger

For nominal and pronoun subtype tagging, we count the frequency for each pair of <nominal, frequency>, and assign the subtype with maximum probability to each test mention. If the nominal does not appear in the training data, we assign it the most common subtype in the training corpus. Since the more accurate subtype tagging for person nominals and pronouns relies on their contexts, we use heuristic rules to disambiguate.

We trained a multi-class tagger to identify name subtypes. The training data consists of entity subtype tagging in ACE 05 training data, and ACE 04 training data with subtypes mapped to 05 subtype set. We augment the tagger by post-processing patterns for high-frequency names including special characters. For example, we assign all organizations ending with “Party” the subtype “Non-governmental”. In addition, we collect country/state/province/city name lists to fix the corresponding name subtypes.

To generate the subtype of an entity from its mentions, we use the following rule: If an entity includes names, assign its subtype as the first name’s subtype, otherwise assign it as the first non-pronoun’s subtype.

2.4 Coreference Resolver

The coreference resolver goes through three successive stages. First, high-precision heuristic rules make some positive and negative reference decisions about mention pairs on the basis of string matching and syntactic analysis. This year we also extract nominal clusters and event patterns to extend these rules. For example, for any two entity mentions M_1 , M_2 appearing as arguments in a event mention, (1) For Business/Life/Movement/Conflict/Contact/Justice/Transaction/Personnel-End-Position events, if M_1 , M_2 are not in an apposition, they are very unlikely to be coreferential; (2) For Personnel (Start-Position, Nominate, Elect) events, if M_1 and M_2 are persons, they are very likely to be coreferential (for example, “Fred” and “president” will be coreferential in “Fred was named president.”).

The remaining pairs are assigned confidence values by a combination of maximum entropy models. Since different mention types have different coreference problems, we separate the system into different maximum entropy classifiers for names, nominals, and pronouns. Each classifier operates on a distinct feature set. We then use the output of the relation tagger to rescore coreference hypotheses and get the final coreference decisions. We will briefly discuss the re-scoring techniques in section 6.

2.5 Relation Tagger

The relation tagger uses a K-nearest-neighbor algorithm. We consider a mention pair as a possible instance of a relation only when: (1) the number of mentions between their heads is less than a threshold (different threshold values for different types), and (2) the coreference probability produced for the pair by the baseline resolver is lower than a threshold. Each training / test example consists of the pair of mentions and the sequence of intervening words. Associated with each training example is either one of the ACE relation types or no relation at all. We defined a distance metric between two examples based on:

- whether the heads of the mentions match
- whether the ACE types of the heads of the mentions match (for example, both are people or both are organizations)
- whether the ACE subtypes of the heads of the mentions match
- whether the intervening words (pruned by chunking information and stop-word list) match

To tag a test example, we find the k nearest training examples (where $k = 3$) and use the distance to weight each neighbor, then select the most common class in the weighted neighbor set.

To provide a crude measure of the confidence of our relation tagger, we define two thresholds, D_{near} and D_{far} . If the average distance d to the nearest neighbors $d < D_{near}$, we consider this a *definite* relation. If $D_{near} < d < D_{far}$, we consider this a *possible* relation. If $d > D_{far}$, the tagger assumes that no relation exists (regardless of the class of the nearest neighbor).

The relation tagger is trained from ACE 05 training data, and ACE 04 data with types/subtypes mapped into 05 set.

3 Research Focus 1: Name Re-Ranking by Later IE Components

In order to improve an information extraction system, we believe it is important to take advantage of more global knowledge. Error analysis on the baseline components also suggested that many errors can be eliminated by applying knowledge produced by later stages. We demonstrate this by using the results of later ACE components, namely coreference analysis, relation extraction, and event patterns, to reduce the errors produced by the name tagger. We applied the basic techniques described in (Ji and Grishman, 05), improving the methods of sample creation and extending the feature set.

3.1 Re-Ranking Model Construction

3.1.1 Hypothesis Representation

In our name re-ranking model, each hypothesis is an NE tagging of the entire sentence. We generate different numbers of candidate hypotheses: $N = 5, 10, 20$ or 30 for different margin ranges, by cross-validation checking the training data for the rank of the best hypothesis for each sentence. With this N , optimal reranking (selecting the best hypothesis among the N best) would yield a weighted F-measure=96.1 on our development set. And we do further pruning by beam setting, removing candidate hypotheses with very low probabilities from the HMM.

3.1.2 Re-Ranking Functions

The goal of our name re-ranking model is to learn a ranking function f of the following form: for each pair of hypotheses (h_i, h_j) , $f : H \times H \rightarrow \{-1, 1\}$, such that $f(h_i, h_j) = 1$ if h_i is better than h_j ; $f(h_i, h_j) = -1$ if h_i is worse than h_j . In this way we are able to convert ranking into a classification problem. And a maximum entropy model for re-ranking these hypotheses can be trained and applied.

During training we use ACE-weighted F-measure to measure the performance of each name hypothesis against the key. For different types of name errors (Entity level/type mismatch), we add weights used in the ACE value metric to count F-measure.

3.1.3 Sample Pruning

If we pick up all hypotheses pairs (h_i, h_j) as samples, the number of samples produced from N-best output will be $O(N^2)$. To make the procedure of both training and test more efficient, we incorporate the sample pruning methods used in parsing and machine translation re-ranking. The methods include the following: 1. Prune the pairs in which the performance/probability of two hypotheses are very close; 2. If the margin between h_i and h_j is large enough, output better/worse directly; 3. During decoding, if one hypothesis h_i is worse than 80% of the other hypotheses, discard it directly; 4. If the probability of the re-ranking decision from maxent for (h_i, h_j) is larger than a threshold, don't consider the results of the pairs $\langle h_j, \text{any other hypothesis} \rangle$.

The training samples are obtained by running the name tagger in cross-validation on the ACE 03, 04 and 05 training data. We used 980,305 samples to train the coreference and relation based re-rankers, while using 121,082 samples from the sentences including event trigger words to train the event pattern based re-ranker.

3.1.4 Incremental Re-Ranking

From each re-ranking step we output the best name hypothesis directly if the re-ranker has high confidence in its decisions. Otherwise the sentence is forwarded to the next re-ranker, based on other features. The goal is to select the best decision for each sentence gradually. In the following we will present the re-rankers using features from three different components separately.

For most features we compute their value for individual name candidates first, and then sum over all names in a hypothesis. Finally we determine, for a given sentence and a pair of hypotheses (h_i , h_j), whether this sum is greater for h_i or h_j . The results of these comparisons are used as features in assessing the ranking of h_i and h_j . In the following we describe the features of the individual re-ranking stages.

3.2 Re-Ranking by Coreference features

As described in (Ji and Grishman, 05), we capture seven coreference features for re-ranking the name hypotheses. The features capture evidence of the correctness of a name provided by reference resolution; for example, a name which is coreferenced with more other mentions is more likely to be correct, so we use the number of mentions corefed to the candidate name, whether the name is the first mention in the entity, etc. We also capture some other local or name-internal evidence; for instance, that an organization name includes an explicit organization-indicating suffix.

3.3 Re-Ranking by Relation features

Following the above first-stage re-ranking by coreference, we proceed to a second-stage re-ranking based on ACE relation information. We kept the following features described in (Ji and Grishman 05): conjunction of “definitely in relation or not” information and margin value; weighted voting value; and the probability score from coreference based re-ranking. In addition we extended the feature set by relation constraints.

From the ACE training corpus, for each mention pair involved in a relation $R(\text{ARG1}=M_i, \text{ARG2}=M_j)$, if M_i is a name, we compute $\text{Prob}(M_i = \text{EntityType}_k \mid M_j = \text{EntityType}_l, R = \text{RelationType}_m)$. Then we scale these probabilities into an additional indicator for M_i being identified as the correct entity type: 1 (very_likely_correct_type), 0.5 (likely_correct_type), 0 (unsure), -0.5 (unlikely_correct_type), -1 (very_unlikely_correct_type). For example, the name M_i matching the condition ($M_i = \text{PER} \mid M_j = \text{PER}, R = \text{Per_Social}$) is very likely to be correct; while ($M_i = \text{ORG} \mid M_j = \text{PER}, R = \text{Per_Social}$) is very unlikely to be correct, etc. This information helps to select the hypothesis with correct name type recognition.

Finally, we collect the high-frequency name lists from the training corpus, and country/province/state/city lists from Chinese wikipedia. If a name candidate is identified as the same type in one of these lists, we increase the confidence for the corresponding hypothesis.

3.4 Re-Ranking by Event features

After the second-stage relation based re-ranking, the sentences will enter the third-stage re-ranking based on event patterns if they have a low ranking confidence from the second-stage re-ranking, and they include an event trigger word.

3.4.1 Event Pattern Extraction

We extract event patterns as follows:

1. Extract all the event mention strings from the ACE05 training corpus, and classify them into eight types according to the different event types.
2. For each event mention string, replace the trigger word by its event type_subtype, and replace each argument by its entity type.
3. Run the POS tagger and chunker on the strings obtained from step 2.
4. Edit the patterns by hand, replacing tokens by their part of speech, chunk type, or a wild card or deleting them entirely if they are not relevant to detecting the event. Some patterns are collapsed, and some patterns which appear too specific or too general are deleted.

To insure that patterns are not over-generalized by the hand editing, the corpus is split in two and patterns derived from one half are, after hand editing, applied to the other half and their accuracy in event prediction is reviewed.

3.4.2 Feature generation based on event patterns

We developed five sets of patterns for personnel, contact, life, business, and conflict events¹. Based on the patterns we trained five third-stage re-ranking models and applied them to the name hypotheses.

For each name hypothesis, we identify the event trigger word and apply one of these five models. We find the event pattern that most closely matches the sentence, ignoring the entity types of the arguments. We then compare the types of the names filling argument slots with the entity types in the pattern and assign a score which is positive if all entity types match, negative if some do not match, and zero if the argument slots are empty.

Besides re-scoring, we developed several heuristic rules based on event pattern information to prune name candidates in the best hypothesis.

4 Research Focus 2: Using Semantic Roles to Improve EMD

From the previous re-ranking steps some name hypotheses still get low ranking confidence. Collecting them together with the ambiguous nominals we get a “difficult multiple mention set” for each document. These mentions need additional semantic information to be disambiguated. We attempted to address this problem by using semantic role information to re-score the tags for these mentions.

4.1 Semantic Role Clustering

The basic idea at this stage is to use selectional preferences at the level of predicate-argument structure to resolve these ambiguous names and nominals. However, acquiring such preferences for each combination of verb and role (ARG0, ARG1, ...) would require a great deal of training data; we needed some way of reducing the space of preferences to be learned. In the frame files in Chinese Propbank (Xue et al., 2005) there are role descriptors for each role, e.g., the ARG0 may have a descriptor like "doer of deed" or "agent" and the ARG1 may have a descriptor like "thing done" or "theme". We can map each combination of predicate and core argument into a role-descriptor. But the role descriptors are still too specific (3380 different role descriptors), so if we use them directly to re-score mention decisions, the feature space is too sparse. Therefore we clustered these roles by string-based heuristics which reduced the number of different role descriptions (for example, by selecting one word from a longer description). We show some examples of these clusters in Table 1.

Predicate	Argument	Semantic Role Description	Semantic Role Cluster
搬 (move)	ARG0	Mover	AGENT
信服 (convince)	ARG0	People-convinced	PEOPLE
辞退 (dismiss)	ARG1	Employee-dismissed	EMPLOYEE
直驶 (drive)	ARG2	Vehicle-arg0-drives/ road-arg0-drives-on	VEHICLE
访问 (visit)	ARG1	place-visited	LOCATION
现任 (currently hold the post)	ARG1	Position-arg0-currently-holds	POSITION

Table 1. Examples of Semantic Role Clusters

This clustering yields 1116 semantic role clusters. Besides these clusters extracted from core arguments, we keep the secondary argument ARGM-LOC as an additional cluster which strongly indicates LOC or GPE types.

¹ We didn't use the movement and justice event patterns because there were too many noisy instantiations, and didn't use the transaction patterns due to the small number of training samples

4.2 Re-scoring Mentions by Semantic Role clusters

4.2.1 Training and Test

Taking the difficult multiple mention sets as input, we train and apply a MaxEnt multi-class classifier to re-predict each mention head's entity type: PER/ORG/GPE/LOC/FAC/NAN. Each training sample is a mention, tagged with its (hand-corrected) semantic role cluster and entity type. We construct training samples for this classifier from two sources:

1. Run the baseline mention tagger on Chinese PropBank, so we get key semantic roles, and output the mentions which are tagged with high-confidence entity types.
2. Run Xue's Chinese semantic role tagger (Xue et al., 05) trained from Chinese PropBank V1.0 on ACE 05 training data; and prune the singletons.

During the test procedure we first run Xue's SRL tagger on the document, then apply the derived re-scoring model.

4.2.2 Feature set

The features used in the re-scoring model include

- Semantic role cluster constraint

For each mention head tagged with $EntityType_i$ and $ClusterSet_k$ in the training corpus (where $ClusterSet_k$ is the role or roles assigned to the mention by predicate(s) in the sentence), we use the probability $Prob(EntityType_i|ClusterSet_k)$.² We scale the values into bins and use them as a feature.

- Semantic role description constraint

By the same method as above, we get the statistics for $Prob(EntityType_i|RoleDescriptor_k)$

- The number of clusters the head belongs to
- Predicate
- Argument type (ARG0, ARG1, ...)
- "Mapitem" in Propbank frame files (subject, object, etc.)
- Conjunction of Predicate and Argument type, Predicate and Mapitem
- Entity type produced by previous components

In addition to feature-based rescoreing, we developed several heuristic rules using the semantic role descriptors to correct entity types.

5 Research Focus 3: Semi-Supervised Learning

We used semi-supervised learning to extend the data for both name tagging and nominal tagging.

5.1 For Name Tagging

Using the LDC ACE05 unannotated data, we do semi-supervised learning in bootstrapping style:

1. Split the data into 10 subsets and mark them C_1, C_2, \dots, C_{10} . Call the updated HMM name tagger (without re-ranking/re-scoring models) NameM. Use a small set of ACE training data as our development test set, DS. Select three thresholds $T_1 > T_2 > T_3$.
2. For $i=1$ to 10
 - (1) Run NameM on C_i ; output the sentences with margin $> T_3$
 - (2) For each tagged sentence S in C_i ,
 - If S does not include any names, or if it has fewer than six words, remove S ;

² A mention can be assigned two roles if it appears as the argument of more than one predicate; for example, in "The man I hired killed Fred." "man" is assigned the EMPLOYEE and AGENT role descriptors by "hired" and "killed", respectively. In such cases we combine the probabilities of entity types given by the two roles.

- Else if margin (of S) $\geq T_1$, keep S.
 Else if ($T_1 > \text{margin} \geq T_2$) or ($T_2 > \text{margin} \geq T_3$ and S includes at least one new name),
 then manually review S
 (it took a native speaker two days to process the 19897 sentences.)
- (3) Relabel the current name tagger (NameM) as OldNameM,
 add C_i to the training data,
 and re-train the name tagger, producing an updated model NameM.
- (4) Run NameM on DS; if the performance gets worse, reset NameM = OldNameM

5.2 For Nominal Tagging

Using the nominal head lists extracted from ACE training data as seeds, we apply ten conjunction patterns on unannotated LDC data to extract more nominal heads. For example, "PERNOM₁ VERB₁, (and) PERNOM₂ VERB₂", if PERNOM₁ appears in the baseline nominal list, we extract PERNOM₂; taking those high-frequency PERNOM₂ plus manual pruning, we extend the nominal list. Then we repeat this procedure by using the updated lists as new seeds. After three iterations we managed to extend all nominal head lists by about 20%. In addition, we extend the person title word list by running the name tagger on the Chinese TreeBank and checking person contexts.

6 Research Focus 4: Re-Scoring Reference Resolution by Relations

For coreference resolution, as described in (Ji et al., 05), we use the output of a relation tagger to encode the relational semantic context of mention pairs into patterns, and rescore coreference hypotheses.

First we apply MaxEnt models without any relation information to produce a probability of coreference for each mention pair. This probability becomes a feature in the rescoring stage of maxent classification, together with features representing the relation patterns. If a high reliability instantiation of one of the patterns applies to a given mention-antecedent pair, we include the following features for that pair: the type of the pattern, the reliability of the rule instantiation, the relation type and subtype, the direction of the relation, and the tokens for the two mentions. Besides that, in this year's system we add one more feature: the distance value from the KNN relation tagger, scaled into 4 bins.

Acknowledgement

We would like to thank Dr. Daniel Bikel for providing the parser package, Prof. Martha Palmer and Dr. Xianwen Xue for providing the Chinese semantic role tagger, and Dr. Min Wan for providing the Chinese segmenter. This research was supported in part by the Defense Advanced Research Projects Agency.

References

- Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance Learning Name-finder. *Proc. Fifth Conf. on Applied Natural Language Processing, Washington, D.C.* pp. 194-201.
- Daniel M. Bikel. 2004. Intricacies of Collins' Parsing Model. *Computational Linguistics* 30(4). pp. 479-511.
- Heng Ji and Ralph Grishman. 2005. Improving Name Tagging by Reference Resolution and Relation Detection. *Proc. ACL2005*. pp. 411-418. Ann Arbor, USA.
- Heng Ji, David Westbrook and Ralph Grishman. 2005. Using Semantic Relations to Refine Coreference Decisions. *Proc. HLT/EMNLP2005*. pp. 17-24. Vancouver, B.C., Canada
- Nianwen Xue and Martha Palmer. 2005. Automatic Semantic Role Labeling for Chinese Verbs, in *Proceedings of the 19th International Joint Conference on Artificial Intelligence. Edinburgh, Scotland.*
- Nianwen Xue, Fei Xia, Fu-Dong Chiou and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207-238.