Bootstrapping a Hidden Markov Model for Relationship Extraction Using Multi-level Contexts*

Cheng Niu, Wei Li, Rohini Srihari, Laurie Crist

Cymfony Inc., 600 Essjay Road, Williamsville, NY 14221, USA
{cniu, wei, rohini, lcrist}@Cymfony.com

This paper presents a new bootstrapping approach to relationship extraction from raw text. The bootstrapping procedure consists of two learning phases. First, symbolic relationship extraction rules are learned after three levels of processing, namely, post-Named-Entity-tagging, post-shallow-parsing, and post-deep-parsing. Then, a Hidden Markov Model (HMM) classifier is trained to identify the targeted relationship from the post-shallow-parsing context. The HMM training utilizes a corpus automatically tagged by the symbolic rules learned in the first phase. The resulting HMM is in effect a generalization of the recognized post-shallow-parsing contexts, hence it achieves higher recall. Benchmarking shows that this new bootstrapping method approaches supervised learning methods in performance. The contributions of different levels of contexts are also evaluated.

Key words: Bootstrapping, Unsupervised Learning, Hidden Markov Model, Relationship Extraction, Information Extraction.

1 INTRODUCTION

Detecting relationships between entities is one major task in Information Extraction (IE) beyond Named Entity (NE) tagging. Relationships are major building blocks for entity profiles (EPs) whose extraction is an intermediate level IE task with great potential in applications (Li *et al.* 2003). The Message Understanding Conference (MUC) has defined three relationships as targets for extraction in its Template Relation (TR) task (Chinchor & Marsh 1998): (i) the LOCATION_OF relationship from an organization (ORG) entity to a location (LOC) entity, (ii) the EMPLOYEE_OF relationship from an ORG entity to a person entity, and (iii) the PRODUCT_OF relationship from an ORG entity to a product entity.

Research on relationship extraction uses various techniques, including handcrafted rules (Aone & Ramos-Santacruz 2000) (Li *et al.* 2003), and supervised machine learning, such as the shallow parsing-based kernel method (Zelenko *et al.* 2002). Rule-based systems and supervised learning systems achieve state-of-the-art performance for relationship extraction. However, both approaches face a serious *knowledge bottleneck*. Rule-based systems require a significant amount of skilled labor. Supervised learning needs a sizable manually truthed training corpus to cover the variety of the contexts representing the relationship. This knowledge bottleneck makes it difficult to do rapid domain porting and to support user-defined relationship extraction. This motivates the relationship extraction research using unsupervised machine learning that only requires a raw corpus from a given domain.

* This work was partly supported by a grant from the Air Force Research Laboratory's Information Directorate (AFRL/IF), Rome, NY, under contract F30602-03-C-0044. The authors wish to thank Carrie Pine of AFRL for supporting and reviewing this work.

Riloff (1996) described a system which automatically generates parsing-based information extraction patterns from an untagged corpus. This system requires document classification. Agichtein & Gravano (2000) proposed a bootstrapping approach for relationship extraction which only requires a few relationship instances (facts)[1] as initial seeds. Ravichandran & Hovy (2002) uses a bootstrapping method that extracts relationships from the web in order to enhance their Question-Answering system. Unlike Zelenko *et al.* (2002), these two bootstrapping systems only use surface text patterns. Due to the lack of parsing support and a pattern generalization mechanism, Ravichandran & Hovy (2002)'s system suffers from limited extraction recall. To improve recall, Agichtein & Gravano (2000) implemented a clustering algorithm to model important features of the targeted surface token sequences.

Bootstrapping approaches have also been explored in Natural Language Processing (NLP)/IE tasks other than relationship extraction. Collins & Singer (1999) used co-training for NE classification. Cucerzan & Yarowsky (1999) described a bootstrapping technique for language-independent NE tagging based on two orthogonal evidence sources, namely, context evidence and morphology evidence. Yarowsky (1995) presented a bootstrapping approach for word sense disambiguation.

This paper presents a new bootstrapping approach to relationship extraction following our successful application of two-step learning to NE classification (Niu *et al.* 2003). It consists of two learning phases. First, precision-oriented symbolic rules are learned at three structural levels: post-Named-Entity-tagging (post-NE), post-shallow-parsing (post-SP), and post-deep-parsing (post-DP). To improve the system recall, a Hidden Markov Model (HMM) is trained to classify whether the incoming post-SP contexts express the targeted relationship. The training of the HMM uses the corpus automatically tagged by the symbolic rules learned in the first learning phase. The novelty of this research lies in three aspects: (i) the exploration of multi-level structural contexts; (ii) the separation of the pattern generalization process from the pattern extraction process: targeted patterns are extracted during bootstrapped iterative learning, and the learned patterns are subsequently generalized by statistical modeling; (iii) formulating the pattern generalization task as a language modeling task.

The remaining text is structured as follows. The system design is presented in Section 2. Section 3 describes the bootstrapped symbolic rule learning using multi-level contexts. Section 4 presents the training of the HMM as a post-SP context classifier. Section 5 shows the experiment procedure and benchmarks, followed by the conclusion in Section 6.

2 SYSTEM DESIGN

2.1 System Architecture

Although an infinite number of relationships can occur between entities, there are certain relations that are more predictable and relatively permanent, with respect to *Temporal Granularity* (Hobbs & Israel 1994), than others. Such relations are targets for relationship extraction, in contrast to event extraction, which aims at capturing more dynamic relations (actions) and the related roles involving entities.

The above nature of relationships between entities is critical for the feasibility of bootstrapped relationship learning from raw text. If a relationship is not subject to constant changes (e.g. LOCATION_OF, BIRTHDAY_OF), the mention of the same entity pair in a sentence usually represents this relationship. The context redundancy in the multiple mentions of a given relationship instance is the key for effective unsupervised relationship learning (Agichtein & Gravano 2000).

[1] Relationship *instance* refers to a unique entity pair that holds a targeted relationship. Note that there may be multiple *mentions* of the same relationship instance in a corpus.

Figure 1 shows the system architecture for our bootstrapped relationship learning. A large raw corpus is first parsed by our *InfoXtract* parser (Srihari *et al.* 2003), which consists of a pipeline of components, mainly, an NE Tagger, Shallow Parser and Deep Parser. The original text and its associated processing results - including NEs, SP units, and DP dependency trees - are stored in an indexed repository which supports high speed retrieval.

The bootstrapping starts with the input of a few seeds in the form of entity pairs that hold the targeted relationship, e.g. {Microsoft, Redmond} for the LOCATION_OF relationship, {[Abraham Lincoln], [February 12, 1809]} for the BIRTHDAY_OF relationship. The learning system then retrieves all sentences containing the entity pairs as contexts. Three levels of context are retrieved: (i) post-NE token sequence; (ii) post-SP unit sequence; and (iii) post-DP dependency trees.

In the symbolic rule learning stage, the patterns are extracted from one of the three levels of contexts, and patterns with high accuracy are learned as relationship extraction rules. These learned rules are applied to the training corpus in the repository to extract new relationship instances. Then the contexts of these new instances are used to learn more rules iteratively. The symbolic rule learning algorithm stops when no new rules can be learned.

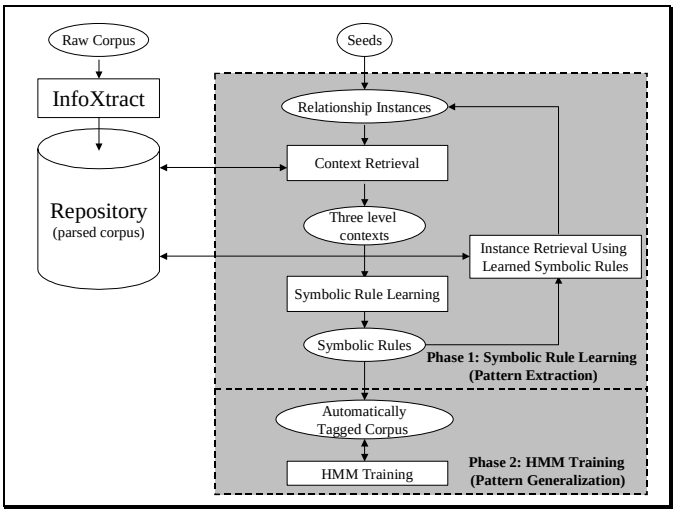

Figure 1. Bootstrapping Architecture for Relationship Extraction

In the second learning phase, the post-SP contexts associated with the recognized relationship instances are used to train an HMM-based context classifier. The resulting HMM is a generalization of the post-SP contextual patterns learned in the previous stage. Therefore, the recall is significantly enhanced.

2.2 Why Multi-level: Linguistic Justification

Before describing the details on how the different levels of contexts are retrieved for relationship learning, it is necessary to present the rationale behind the multi-level learning. In order to capture the linguistic phenomena representing relationships, multi-level rules are desirable since relationships between entities are often expressed in English at different levels of linguistic structure (Li *et al.* 2003). Three levels of structure are identified for effective relationship extraction via pattern matching.

The first level is NE, the results are a linear token string except for multi-token NEs which have been combined into a single compound token. Relationship rules learned at this level are designed to capture surface level phenomena. For example, the linear pattern *LOC -based ORG*

will match cases like *Seattle-based Microsoft.* For this type of very local phenomena, linguistic parsing beyond NE is not helpful.

The second level is Shallow Parsing (SP). SP aims at grouping the basic linguistic units, mainly, *BaseNP* (Basic Noun Phrase) and *VG* (Verb Group) (Church 1988). The results create a basic structural foundation for capturing some relationship patterns. The following is a pattern rule for the relationship EMPLOYEE_OF: *PER , position_word_list of/for/from ORG.* This rule will handle cases like *Robert Callahan, spokesman of Seattle-based Microsoft.* In this example, the relationship EMPLOYEE_OF from *Microsoft* to *Robert Callahan* cannot be captured without the SP structural basis which allows pattern matching to 'jump over' pre-modifiers like *Seattle-based*.

The third level is Deep Parsing (DP). DP in InfoXtract is designed to decode underlying logical dependency relationships, in effect a type of *logical form.* It is especially suited for targeting relationships expressed by logical Subject-Verb-Object (SVO) structures. The following sample is a DP-based pattern rule for the relationship EMPLOYEE_OF.

'work for' Subject: PER
Object: ORG

This rule is able to cover cases like *Robert Jackson originally from Washington, D.C. has been working for Seattle-based Microsoft for almost a decade.* In the above example, the adverb *originally* and the prepositional phrase *from Washington, D.C.* no longer stand in the way of pattern matching as DP produces a dependency link directly between *work for* and *Robert Jackson*.

2.3 HMM as a Context Classifier

The rationale behind using an HMM for relationship extraction lies in the equivalence of the relationship extraction task to the binary context classification task. That is, given a context containing a pair of appropriate entities, determine whether the context expresses the targeted relationship.

Like the post-NE context, the post-SP context is also a linear token sequence. So the context classification for relationship detection is equivalent to the binary token sequence classification. This is a language modeling task (Jelinek 1997). HMM is generally recognized as one of the most powerful devices for language modeling (Murphy 1995). One important feature of probabilistic finite automata, such as HMM, is the feasibility of learning from positive instances only (Murphy 1995). This feature makes this new bootstrapping procedure possible since the symbolic rules learned from the first phase can only provide positive instances.

Agichtein & Gravano (2000) uses a context clustering method based on a vector space model for pattern generalization. We believe that the induction of an HMM as a context classifier is a more effective approach since word order is taken into account in addition to uni-gram statistics.

3 SYMBOLIC RULE LEARNING

3.1 Context Retrieval

Three context representations are explored corresponding to the three levels of InfoXtract processing. The post-NE context consists of a linear sequence of words and identified NEs. The post-SP context is represented as a linear sequence of linguistic units, i.e. words (e.g. function words, or words which failed to be constructed), NEs and basic phrases constructed by SP. In the case of basic phrases, the units are represented by their head tokens, which will be illustrated shortly. One exception is the noun phrase which contains at least one targeted entity in a pre-modifier or specifier position (e.g. *Japan* in NP[*Japan's Nissan Motor*], *IBM* in NP[*IBM's headquarters*]). In this case, the noun phrase context is represented in the same way as the post-

NE context while the surrounding context remains at the post-SP level. The post-DP context is defined as the minimum parsing tree that contains the targeted entity pair.

The following example illustrates the multi-level processing of our system, with the three-level context representations for the targeted entity pair {Japan, [Nissan Motor]}:

(1) Input sentence: *Japan's Nissan Motor will build a $1 billion state-of-the-art auto plant.*

(2) Post-NE context: *NE[Japan] 's NE[Nissan Motor] will build a NE[$ 1 billion] state-of-the -art auto plant .*

(3) SP results: *NP[[Japan] 's [Nissan Motor]] VG[will build] NP[a [$ 1 billion] state-of-the-art auto plant] .*

(4) Post-SP context: *Japan 's [Nissan Motor] build plant*

(5) DP results: *[will build]* Subject: *[Nissan Motor]*
Modifier: [*Japan*]
Object: [*state-of-the-art auto plant*]
Modifier: [*$ 1 billion*]

(6) Post-DP context: [*Nissan Motor*]
Modifier: [*Japan*]

3.2 Symbolic Rule Learning

The symbolic rules have the following format:

Rule =: RuleLevel RulePattern
RuleLevel =: @post-NE | @post-SP | @post-DP

For @post-NE rules or @post-SP rules, *RulePattern* is a linear sequence containing the targeted entity pair. For each context in the form of $a_0\ a_1\ @\text{NeX}\ a_2 \cdots a_3\ @\text{NeY}\ a_4\ a_5$, three rule patterns are extracted in rule learning (@NeX and @NeY are the targeted entity pair):

$@\text{NeX}\ a_2 \cdots a_3\ @\text{NeY}$

$a_1\ @\text{NeX}\ a_2 \cdots a_3\ @\text{NeY}\ a_4$

$a_0 a_1\ @\text{NeX}\ a_2 \cdots a_3\ @\text{NeY}\ a_4 a_5$

Assuming NeX=LOC and NeY=ORG, in rule learning, @LOC is a symbol for "checking whether the current word is an identified NE of LOC type", and @ORG stands for "checking an identified NE of ORG type". Other items in the pattern are based on string matching where string refers to the canonical form of the token, after orthographic and morphology normalization, e.g. Gone/GOING/go/goes/went/gone → go. For @post-DP rules, RulePattern checks along the logical dependency links instead of linear neighboring tokens, e.g. logical Subject, logical Object, Complement, Modifier, etc.

Given the following sentence containing the targeted entity pair {ORG: [*Boeing Corporation*], LOC: [*Chicago*]}: *The announcement that [Boeing Corporation] is moving its worldwide headquarters to [Chicago] has drawn praise from Gov. George Ryan,* the following seven rule candidates are constructed:

@post-NE @ORG *is moving its worldwide headquarters to* @LOC

@post-NE *that* @ORG *is moving its worldwide headquarters to* @LOC *has*

@post-NE announcement *that* @ORG *is moving its worldwide headquarters to* @LOC *has drawn*

@post-SP @ORG *moving headquarters to* @LOC

@post-SP *that* @ORG *moving headquarters to* @LOC *drawn*

@post-SP *announcement that* @ORG *moving headquarters to* @LOC *drawn praise*

@post-DP *move*
Subject*:* @ORG
Object: *headquarter*
Complement: PP[*to* @LOC].

To evaluate the accuracy of each rule candidate, we define the positive mentions and negative mentions of each rule candidate as follows: positive mentions are the existing relationship mentions recognized by the rule candidates. Negative mentions are defined as the mentions in conflict with the existing recognized relationships. As discussed before, this is a conservative definition to help guide the precision-oriented rule learning. This definition will treat any relationship mentions which are unlisted in the current instance set as negative mentions. The accuracy of a rule candidate is computed using Equation 1.

$$\text{Equation 1. accuracy} = \frac{\text{positive mentions}}{\text{positive mentions} + \text{negative mentions}}.$$

The algorithm for 3-level symbolic rule learning is as follows:

Step 1. Provide relationship seeds for bootstrapping, and put them into the relationship instance set;
Step 2. Retrieve 3-level contexts from the repository for each new instance;
Step 3. Construct 3-level rule candidates based on the retrieved contexts; evaluate the accuracy of each rule candidate: rule candidates with accuracy >= 0.9 and positive mentions >= 2 are learned and sent into symbolic rule set;
Step 4. Stop if no new symbolic rules are learned; otherwise, apply the newly learned rules to the corpus in the repository, and put the extracted relationship instances into the relationship instance set, go to Step 2.

4 HMM TRAINING FOR RELATIONSHIP EXTRACTION

The symbolic rules are in the form of an exact token sequence match or an exact parsing tree match. These rules suffer from low recall (benchmarked as 38% in Table 2). To increase the recall, we use an HMM to generalize the token sequence patterns.

The HMM training is based on a corpus automatically tagged by applying the learned rules. All recognized post-SP contexts are retrieved from the repository to form a training corpus for the HMM. It has been observed that the post-SP level output provides the optimum support for HMM training; its basic phrase structures facilitate longer contextual checks.

Each post-SP context may begin up to two tokens before the targeted entity pair and end up to two tokens following the entity pair, i.e. $a_0\ a_1$ @NeX $a_2 \cdots a_3$ @NeY $a_4\ a_5$. The following are training samples for a seeded entity pair {HP, [Palo Alto, Calif.]} for the LOCATION_OF relationship. Note that @LOC and @ORG are treated as two special token symbols in the patterns.

@ORG , based in @LOC , revised
Based in @LOC , @ORG won award
medium , @LOC - base @ORG save $1,000,000
……

Given the above post-SP context corpus, a bi-gram HMM is used to estimate the generation probability of any token sequence as a post-SP context expressing the targeted relationship. To handle tokens with low frequency, each token is associated with one of the following single token features:

twoDigitNum, fourDigitNum, containsDigitAndAlpha, containsDigitAndDash, containsDigitAndSlash, containsDigitAndComma, containsDigitAndPeriod, otherNum, allCaps, capPeriod, initCap, lowerCase, other.

The definitions of these features are the same as (Bikel 1997). The bi-gram HMM is defined as follows. For any token sequence $W = \langle w_0 f_0 \rangle \cdots \langle w_n f_n \rangle$ (where f_j denotes a single token feature defined above), the generation probability of this token sequence as a relevant post-SP context is computed using Equation 2. Equations 3 through 6 represent the HMM back-off model.

Equation 2. $\Pr(W) = \prod_i \Pr(\langle w_i, f_i \rangle | \langle w_{i-1}, f_{i-1} \rangle)$.

Equation 3. $\Pr(\langle w_i, f_i \rangle | \langle w_{i-1}, f_{i-1} \rangle) = \lambda_0 P_0(\langle w_i, f_i \rangle | \langle w_{i-1}, f_{i-1} \rangle) + (1 - \lambda_0) \Pr(\langle w_i, f_i \rangle | f_{i-1})$

Equation 4. $\Pr(\langle w_i, f_i \rangle | f_{i-1}) = \lambda_1 P_0(\langle w_i, f_i \rangle | f_{i-1}) + (1 - \lambda_1) \Pr(\langle w_i, f_i \rangle)$

Equation 5. $\Pr(\langle w_i, f_i \rangle) = \lambda_2 P_0(\langle w_i, f_i \rangle) + (1 - \lambda_2) \Pr(w_i) P_0(f_i)$

Equation 6. $\Pr(w_i) = \lambda_3 P_0(w_i) + (1 - \lambda_3) \frac{1}{V}$

In the above equations, V denotes the size of the vocabulary, the back-off coefficients λ's are determined using the Witten-Bell smoothing algorithm, and the quantities $P_0(\langle w_i, f_i \rangle | \langle w_{i-1}, f_{i-1} \rangle)$, $P_0(\langle w_i, f_i \rangle | f_{i-1})$, $P_0(\langle w_i, f_i \rangle)$, $P_0(f_i)$ and $P_0(w_i)$ are computed by the maximum likelihood estimation. Furthermore, the perplexity of the token sequence W is defined as $PP = 2^{-\frac{\log \Pr(W)}{|W|}}$.

In the tagging stage, for each candidate entity pair, the $\Pr(W)$ of the corresponding post-SP context W is computed. Only when the associated perplexity of the context is lower than a predefined threshold, will the context be recognized as expressing a positive mention of the relationship.

5 EXPERIMENTS AND BENCHMARKING

We used LOCATION_OF in our relationship bootstrapping experiment. Before the iterative learning starts, a large raw corpus (1.2GB, with 88,000,000 words of general news articles) is processed by our parser. The parsing results and the raw text are saved into the repository, which supports high-speed context retrieval (see Figure 1).

The following four entity-pair seeds for the LOCATION_OF relationship were used in our experiment:

{*Microsoft, Seattle*}
{*Microsoft, Redmond*}
{*IBM, Armonk*}
{*Office Depot, Delray Beach*}

In the parsed corpus, 172,575 candidate sentences contain at least one mention of ORG and one mention of LOC. Using the above four seeds, the system learned 3,818 symbolic rules before it stopped learning new rules after 14 iterations. Among the 3,818 rules, there are 1,845 post-NE rules, 1,848 post-SP rules and 125 post-DP rules. These rules extracted 7,645 unique LOCATION_OF relationship instances from the candidate sentences. Some sample rules are shown below:

@post-NE @ORG , in @LOC
@post-SP @LOC – base @ORG
@post-DP
headquarter
Object*:* @ORG
Complement: PP[*in* @LOC].

Finally, an HMM classifier is trained using the contexts recognized by the learned rules.

To evaluate the system performance, two ways of benchmarking are defined: *Retrieval Performance* and *Extraction Performance*. Following Agichtein & Gravano (2000), the Precision (P), Recall (R) and F-score (F) measures for Retrieval Performance are based on counting relationship *instances* while the measures for Extraction Performance are based on counting the *mentions* of each relationship instance (see Footnote 1). Note that from information users' perspectives, the Retrieval Performance is a more meaningful benchmark of the system since it directly reflects the capability for extracting unique relationships. The benchmarking procedure, details and discussion are described below.

Manual benchmarking was performed by our tester. To evaluate the Retrieval performance, 1,000 recognized relationship instances were randomly selected for checking the precision while 1,000 relationship instances whose entity pairs are mentioned at least once within a sentence in the corpus were selected for checking the recall. The precision for symbolic rules is 88%. As for recall, among the 1,000 testing pairs, 378 were extracted by the learned rules, thus the Retrieval Recall for the first-phase symbolic rule learner is 38%.

The precision errors are found to be caused more frequently by the underlying NE tagger rather than by the relationship extractor itself, as shown in Table 1. In other words, assuming a perfect NE input, the bootstrapped relationship extraction system can achieve a precision measure as high as 95.7%.

Table 1. Retrieval Precision of Symbolic Rules

Extracted relationship instances	1,000
Correct	884
Error due to NE Tagging	73
Error due to Relationship Extraction	43
Retrieval Precision	**88%**

Table 2. Multi-level Impact on Retrieval Recall

Post-NE only	19%
Post-SP only	15%
Post-DP only	0%
Post-NE + Post-SP	37%
Post-NE + Post-SP + Post-DP	**38%**

We have also evaluated the contributions of the combinations of individual levels of symbolic rules (Table 2). Findings show that the two shallow levels of contexts (post-NE and post-SP) contribute to the majority of the extracted relationships. The zero recall in only using the post-DP contexts is not surprising: there are no observed recurring post-DP patterns associated with the four seeds, thus no post-DP patterns can survive in the iterative rule learning. The learning simply stops in the first iteration. However, when the post-DP patterns are learned with the post-NE patterns and the post-SP patterns together, relationship mentions tagged by the post-NE or post-SP rules will trigger richer post-DP contexts for the next round of learning. That is why the post-DP contexts still contribute to the overall system recall. For the same reason, the recall of the combined post-NE and post-SP learning is higher than the summation of their respective recall measures. These benchmarks illustrate the benefits of utilizing multiple level contexts in the iterative learning.

It seems to be surprising to see lower recall from the post-SP learning than that from the post-NE learning because the post-SP context is more abstract than the post-NE context and is therefore expected to achieve higher recall. Further study shows that this is because the symbolic rule learning in this approach favors post-NE rules which are more specific than post-SP rules. In other words, the post-NE patterns are more likely to survive than the post-SP patterns in rule learn-

ing. Note that the relationship is not a one-to-one mapping from real world fact to language expressions. For example, the location of the headquarters of *Microsoft* could be expressed in natural language at different levels of geographic granularity, such as *Redmond*, *Seattle*, *Washington*, or even *U.S.* Such variations cannot be fully covered by seeds or the iteratively recognized instances. As a result, the uncovered cases become noise in the learning. For example, assume that *Washington State* is not listed as LOCATION_OF *Microsoft* in the current instance set, then, there is no way for the system to distinguish this relationship mention from negative relationship mentions. In calculating the accuracy of a candidate rule, any relationship mentions which do not directly correspond to relationships in the current instance set are regarded as negative mentions. This conservative scheme favors specific rules and results in more rules learned when using only the post-NE contexts than using only the post-SP contexts.

To evaluate the generalization capability of the HMM, the HMM-based context classifier was trained based on the post-SP contexts associated with the 378 extracted relationship instances. For each candidate relationship instance, all the post-SP contexts associated with the instance are retrieved from the repository, and then classified by the HMM context classifier. If one of the contexts is assigned a perplexity lower than the threshold, that mention is recognized as a targeted relationship. The HMM context classifier extracted 517 out of the 1,000 testing instances. Thus the Retrieval Recall for the second-phase HMM is 52%, an increase of 14% from the first phase. Table 3 shows that HMM is indeed an effective pattern generalizer, having significantly enhanced the recall (+14%) with limited precision loss (-3%).

Table 3. HMM Enhancement for Retrieval

	Symbolic Rules	HMM	Difference
P	88%	85%	-3%
R	38%	52%	+14%
F	53%	65%	+12%

Table 4. Overall Performance Summary

	Retrieval	Extraction
P	85%	82%
R	52%	69%
F	65%	75%

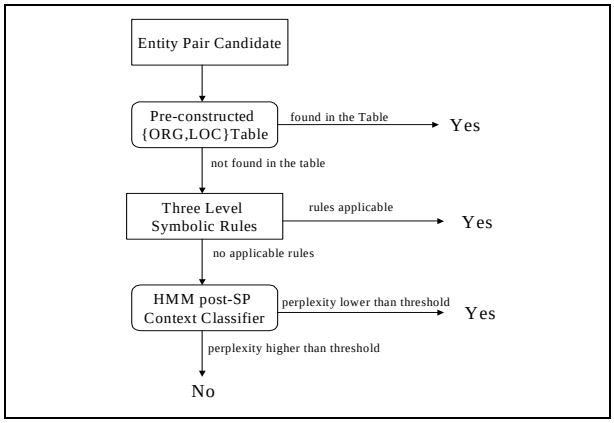

Figure 2. Procedure of Relationship Extraction

In evaluating the Extraction Performance, we built a relationship extraction module that consists of three components: (i) a table containing all the {ORG, LOC} pairs extracted from the corpus; (ii) the symbolic rule model; (iii) the HMM-based post-SP context classifier. We processed a testing corpus by the InfoXtract parser and extracted 50,000 sentences containing LOC and ORG NEs. The extracted 50,000 sentences were then processed using the extraction procedure in Figure 2. The Extraction Precision was calculated by checking 1,000 extracted relationship *mentions* which were selected randomly from the relationship extraction results. For

Extraction Recall, we selected the first 1,000 sentences that contain the LOCATION_OF relationship. The results are shown in the 'Extraction' Column of Table 4. The reason the Extraction Recall is higher than the Retrieval Recall is due to the fact that the common {ORG, LOC} pairs are already extracted in bootstrapping and stored in the pre-constructed table. The relationship instances missed in the table are usually mentioned infrequently.

Zelenko *et al.* (2002) reports an F-score of 83% for LOCATION_OF relationship using kernel-based supervised machine learning. Our bootstrapping method achieves an F-score of 75% which is approaching the performance of supervised machine learning. Our performance is also close or comparable to the best score reported in MUC (75.6% F-score for MUC-7 TR).

6 CONCLUSION

A new bootstrapping approach to correlated entity relationship extraction is presented. The bootstrapping procedure is implemented as two training phases. First, symbolic rules are learned at three structural levels: post-Named-Entity-tagging, post-shallow-parsing, and post-deep-parsing. Then, an HMM is trained to classify whether a context expresses the targeted relationship. The training of the HMM uses the corpus automatically tagged by the symbolic rules learned in the first phrase. The resulting HMM is a generalization of these symbolic rules, hence higher recall is achieved. Benchmarking shows that the performance of the resulting system approaches supervised learning methods.

The performance contributions for combinations of different levels of contexts are evaluated. It is found that the two shallow levels of contexts (post-NE and post-shallow-parsing) contribute to the majority of the extracted relationships.

REFERENCES

Agichtein, E. & Gravano, L. 2000. Snowball: Extracting Relations from Large Plain-Text Collections. *Proceedings of the 5th ACM International Conference on Digital Libraries*. San Antonio.

Aone, A. & M. Ramos-Santacruz 2000. REES: A Large-Scale Relation and Event Extraction System. *Proceedings of ANLP-NAACL 2000*, Seattle.

Bikel, D. M. 1997. Nymble: a high-performance learning name-finder. *Proceedings of the Fifth Conference on ANLP*: 194-201, Morgan Kaufmann Publishers.

Church, K. 1988. A stochastic parts program and noun phrase parser for unrestricted text. *Proceedings of ANLP'88*.

Collins, M. and Y. Singer. 1999. Unsupervised Models for Named Entity Classification. *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC.*

Cucerzan, S. and D. Yarowsky. 1999. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC:*.

Hobbs, J.R. & D. Israel 1994. Principles of Template Design. *Proceedings of Human Language Technology Workshop*: 177-181, NJ.

Jelinek, F. 1997. Statistical Methods for Speech Recognition. The MIT Press.

Li, W., R. Srihari, C. Niu & X. Li 2003. Entity Profile Extraction from Large Corpora. *Proceedings of PACLING'03*. Halifax, Nova Scotia, Canada.

Murphy, K. P. 1996. Passively Learning Finite Automata. Technical Report, Santa Fe Institute.

Niu, C., W. Li, J. Ding & R. Srihari 2003. A Bootstrapping Approach to Named Entity Classification Using Successive Learners. *ACL-2003*. Sapporo, Japan.

Ravichandran, D. & E. Hovy, 2002. Learning surface text patterns for a Question Answering System. *ACL-2002*.

Riloff, E. 1996. Automatically Generating Extraction Patterns from Untagged Text. *AAAI-96:* 1044-1049.

Srihari, R., W. Li, C. Niu & T. Cornell. 2003. InfoXtract: A Customizable Intermediate Level Information Extraction Engine. *HLT-NAACL03 Workshop on The Software Engineering and Architecture of Language Technology Systems* (SEALTS). Edmonton, Canada.

Yarowsky, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. *ACL-1995*.

Zelenko, D., C. Aone & A. Richardella. 2002. Kernel Methods for Relation Extraction. *EMNLP-2002*.