

Singapore Management University Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

6-2003

Using Support Vector Machines for Terrorism Information Extraction

Aixin SUN

Nanyang Technological University, Singapore

Myo-Myo NAING

Nanyang Technological University, Singapore

Ee Peng LIM


Singapore Management University, eplim@smu.edu.sg

Wai LAM

Chinese University of Hong Kong

DOI: https://doi.org/10.1007/3-540-44853-5_1

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the [Databases and Information Systems Commons](#), and the [Numerical Analysis and Scientific Computing Commons](#)

Citation

SUN, Aixin; NAING, Myo-Myo; LIM, Ee Peng; and LAM, Wai. Using Support Vector Machines for Terrorism Information Extraction. (2003). *Intelligence and Security Informatics: First NSF/NIJ Symposium, ISI 2003, Tucson, AZ, USA, June 2-3, 2003: Proceedings*. 2665, 1-12. Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/960

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Using Support Vector Machines for Terrorism Information Extraction^{*}

Aixin Sun¹, Myo-Myo Naing¹, Ee-Peng Lim^{1**}, and Wai Lam²

¹ Centre for Advanced Information Systems, School of Computer Engineering
Nanyang Technological University, Singapore 639798, Singapore
aseplim@ntu.edu.sg

² Department of Systems Engineering and Engineering Management
Chinese University of Hong Kong, Shatin, New Territories, Hong Kong SAR
wlam@se.cuhk.edu.hk

Abstract. Information extraction (IE) is of great importance in many applications including web intelligence, search engines, text understanding, etc.. To extract information from text documents, most IE systems rely on a set of *extraction patterns*. Each extraction pattern is defined based on the syntactic and/or semantic constraints on the positions of desired entities within natural language sentences. The IE systems also provide a set of *pattern templates* that determines the kind of syntactic and semantic constraints to be considered. In this paper, we argue that such pattern templates restricts the kind of extraction patterns that can be learned by IE systems. To allow a wider range of context information to be considered in learning extraction patterns, we first propose to model the content and context information of a candidate entity to be extracted as a set of features. A classification model is then built for each category of entities using Support Vector Machines (SVM). We have conducted IE experiments to evaluate our proposed method on a text collection in the terrorism domain. From the preliminary experimental results, we conclude that our proposed method can deliver reasonable accuracies.

Keywords: Information extraction, terrorism-related knowledge discovery.

1 Introduction

1.1 Motivation

Information extraction (IE) is a task that extracts relevant information from a set of documents. IE techniques can be applied to many different areas. In the intelligence and security domains, IE can allow one to extract terrorism-related information from email messages, or identify sensitive business information from

^{*} This work is partially supported by the SingAREN 21 research grant M48020004.

^{**} Dr. Ee-Peng Lim is currently a visiting professor at Dept. of SEEM, Chinese University of Hong Kong, Hong Kong, China.

news documents. In some cases where perfect extraction accuracy is not essential, automated IE methods can replace the manual extraction efforts completely. In other cases, IE may produce the first-cut results reducing the manual extraction efforts.

As reported in the survey by Muslea [9], the IE methods for free text documents are largely based on *extraction patterns* specifying the syntactic and/or semantic constraints on the positions of desired entities within sentences. For example, from the sentence, “Guerrillas attacked the 1st infantry brigade garrison”, one can define the extraction pattern $\langle \text{subject} \rangle$ **active-attack** to extract “Guerrillas” as a perpetrator, and **active-attack** $\langle \text{direct object} \rangle$ to extract “1st infantry brigade garrison” as a victim³. The extraction pattern definitions currently used are very much based on some pre-defined pattern templates. For example, in AutoSlog [12], the above $\langle \text{subject} \rangle$ **active-attack** extraction pattern is an instantiation of the $\langle \text{subject} \rangle$ **active-verb** template. While pattern templates reduce the combinations of extraction patterns to be considered in rule learning, they may potentially pose as the obstacles to derive other more expressive and accurate extraction patterns. For example, **IBM acquired** $\langle \text{direct-object} \rangle$ is a very pertinent extraction pattern for extracting company information but cannot be instantiated by any of the 13 AutoSlog’s pattern templates. Since it will be quite difficult to derive one standard set of pattern templates that works well for any given domain, IE methods that do not rely on templates will become necessary.

In this paper, we propose the use of Support Vector Machines (SVMs) for information extraction. SVM was proposed by Vapnik [16] and has been widely used in image processing and classification problems [5]. The SVM technique finds the best surface that can separate the positive examples from negative ones. Positive and negative examples are separated by the maximum margin measured by a normal vector w . SVM classifiers have been used in various text classification experiments [2, 5] and have been shown to deliver good classification accuracy.

When SVM classifiers are used to solve an IE problem, two major research challenges must be considered.

- *Large number of instances*: IE for free text involves extracting from document sentences target entities (or instances) that belong to some pre-defined semantic category(ies). A classification task, on the other hand, is to identify candidate entities from the document sentences, usually in the form of noun phrases or verb phrases, and assign each candidate entity to zero, one or more pre-defined semantic category. As large number of candidate entities can potentially be extracted from document sentences, it could lead to overheads in both learning and classification steps.
- *Choice of features*: The success of SVM very much depends on whether a good set of features is given in the learning and classification steps. There should be adequate features that distinguish entities belonging to a semantic category from those outside the category.

³ Both extraction patterns have been used in the AutoSlog system [12].

In our approach, we attempt to establish the links between the semantic category of a target entity with its syntactic properties, and reduce the number of instances to be classified based on their syntactic and semantic properties. A natural language parser is first used to identify the syntactic parts of sentences and only those parts that are desired are used as candidate instances. We then use both the content and syntax of a candidate instance and its surrounding context as features.

1.2 Research Objectives and Contributions

Our research aims to develop new IE methods that use classification techniques to extract target entities, while not using pattern templates and extraction patterns. Among the different types of IE tasks, we have chosen to address the template element extraction (TE) task which refers to extracting entities or instances in a free text that belong to some semantic categories⁴. We apply our new IE method on free documents in the terrorism domain. In the terrorism domain, the semantic categories that are interesting include victim, perpetrator, witness, etc..

In the following, we summarize our main research contributions.

- *IE using Support Vector Machines (SVM)*: We have successfully transformed IE into a classification problem and adopted SVM to extract target entities. We have not come across any previous papers reporting such an IE approach. As an early exploratory research, we only try to extract the entities falling under the **perpetrator** role. Our proposed IE method, nevertheless, can be easily generalized to extract other types of entities.
- *Feature selection*: We have defined the *content* and *context* features that can be derived for the entities to be extracted/classified. The content features refer to words found in the entities. The context features refer to those derived from the sentence constituents surrounding the entities. In particular, we propose the a weighting feature scheme to derive context features for a given entity.
- *Performance evaluation*: We have conducted experiments on the MUC text collection in the terrorism domain. In our preliminary experiments, the SVM approach to IE has been shown to deliver performance comparable to the published results by AutoSlog, a well known extraction pattern-based IE system.

1.3 Paper Outline

The rest of the paper is structured as follows. Section 2 provides a survey of the related IE work and distinguishes our work from them. Section 3 defines our IE problem and the performance measures. Our proposed method is described in Section 4. The experimental results are given in Section 5. Section 6 concludes the paper.

⁴ The template element extraction (TE) task has been defined in the Message Understanding Conference series (MUC) sponsored by DARPA [8].

2 Related Work

As our research deals with IE for free text collections, we only examine related work in this area. Broadly, the related work can be divided into extraction pattern-based and non-extraction pattern-based. The former refers to approaches that first acquire a set of extraction patterns from the training text collections. The extraction patterns use the syntactic structure of a sentence and semantic knowledge of words to identify the target entities. The extraction process is very much a template matching task between the extraction patterns and the sentences. The non-extraction pattern-based approach are those that use some machine learning techniques to acquire some *extraction models*. The extraction models identify target entities by examining their feature mix that includes those based on syntactics, semantics and others. The extraction process is very much a classification task that involves accepting or rejecting an entity (e.g. word or phrase) as a target entity.

Many extraction pattern-based IE approaches have been proposed in the Message Understanding Conference (MUC) series. Based on 13 pre-defined pattern templates, Riloff developed the AutoSlog system capable of learning extraction patterns [12]. Each extraction pattern consists of a *trigger word* (a verb or a noun) to activate its use. AutoSlog also requires a manual filtering step to discard some 74% of the learned extraction patterns as they may not be relevant. PALKA is another representative IE system that learns extraction patterns in the form of *frame-phasal pattern structures* [7]. It requires each sentence to be first parsed and grouped into multiple simple clauses before deriving the extraction patterns.

Both PALKA and AutoSlog require the training text collections to be tagged. Such tagging efforts require much manual efforts. AutoSlog-TS, an improved version of AutoSlog, is able to generate extraction patterns without a tagged training dataset [11]. An overall F_1 measure of 0.38 was reported for both AutoSlog and AutoSlog-TS for the entities in perpetrator, and around 0.45 for victim and target object categories in the MUC-4 text collection (terrorism domain). Riloff also demonstrated that the best extraction patterns can be further selected using bootstrapping technique [13].

WHISK is an IE system that uses extraction patterns in the form of regular expressions. Each regular expression can extract either single target entity or multiple target entities [15]. WHISK has been experimented on the text collection under the management succession domain. SRV, another IE system, constructs first-order logical formulas as extraction patterns [3]. The extraction patterns also allow relational structures between target entities to be expressed.

There have been very little IE research on non-extraction pattern based approaches. Freitag and McCallum developed an IE method based on Hidden Markov models (HMMs), a kind of probabilistic final state machines [4]. Their experiments showed that the HMM method outperformed the IE method using SRV for two text collections in the seminar announcements and corporate acquisitions domains.

TST1-MUC3-0002
 SAN SALVADOR, 18 FEB 90 (DPA) -- [TEXT] HEAVY FIGHTING WITH AIR SUPPORT RAGED LAST NIGHT IN NORTHWESTERN SAN SALVADOR WHEN MEMBERS OF THE FARABUNDO MARTI NATIONAL LIBERATION FRONT [FMLN] ATTACKED AN ELECTRIC POWER SUBSTATION. ACCORDING TO PRELIMINARY REPORTS, A SOLDIER GUARDING THE SUBSTATION WAS WOUNDED.
 THE FIRST EXPLOSIONS BEGAN AT 2330 [0530 GMT] AND CONTINUED UNTIL EARLY THIS MORNING, WHEN GOVERNMENT TROOPS REQUESTED AIR SUPPORT AND THE GUERRILLAS WITHDREW TO THE SLOPES OF THE SAN SALVADOR VOLCANO, WHERE THEY ARE NOW BEING PURSUED.
 THE NOISE FROM THE ARTILLERY FIRE AND HELICOPTER GUNSHIPS WAS HEARD THROUGHOUT THE CAPITAL AND ITS OUTSKIRTS, ESPECIALLY IN THE CROWDED NEIGHBORHOODS OF NORTHERN AND NORTHWESTERN SAN SALVADOR, SUCH AS MIRALVALLE, SATELITE, MONTEBELLO, AND SAN RAMON. SOME EXPLOSIONS COULD STILL BE HEARD THIS MORNING.
 MEANWHILE, IT WAS REPORTED THAT THE CITIES OF SAN MIGUEL AND USULUTAN, THE LARGEST CITIES IN EASTERN EL SALVADOR, HAVE NO ELECTRICITY BECAUSE OF GUERRILLA SABOTAGE ACTIVITY.

Fig. 1. Example Newswire Document

Research on applying machine learning techniques on name-entity extraction, a subproblem of information extraction, has been reported in [1]. Baluja et al proposed the use of 4 different types of features to represent an entity to extracted. They are the *word-level features*, *dictionary features*, *part-of-speech tag features*, and *punctuation features* (surrounding the entity to be extracted). Except the last feature type, the other three types of features are derived from the entities to be extracted.

To the best of our knowledge, our research is the first that explores the use of classification techniques in extracting terrorism-related information. Unlike [4], we represent each entity to be extracted as a set of features derived from the syntactic structure of the sentence in which the entity is found, as well as the words found in the entity.

3 Problem Definition

Our IE task is similar to the template element (TE) task in the Message Understanding Conference (MUC) series. The TE task was to extract different types of target entities from each document, including perpetrators, victims, physical-targets, event locations, etc.. In MUC-4, a text collection containing newswire documents related to terrorist events in Latin America was used as the evaluation dataset. An example document is shown in Figure 1.

In the above document, we could extract several interesting entities about the terrorist event, namely location (“SAN SALVADOR”), perpetrator (“MEMBERS OF THE FARABUNDO MARTI NATIONAL LIBERATION FRONT [FMLN]”), and victim (“SOLDIER”). The MUC-4 text collection consists of a training set (with 1500 documents and two test sets (each with 100 documents)). For each document, MUC-4 specifies for each semantic category the target entity(ies) to be extracted.

In this paper, we choose to focus on extracting target entities in the **perpetrator** category. The input of our IE method consists of the training set (1500 documents) and the perpetrator(s) of each training documents. The training documents are not tagged with the perpetrators. Instead, the perpetrators are stored in a separate file known as the *answer key file*. Our IE method therefore has to locate the perpetrators within the corresponding documents. Should a perpetrator appear in multiple sentences in a document, his or her role may be obscured by features from these sentences, making it more difficult to perform extraction.

Once trained, our IE method has to extract perpetrators from the test collections. As the test collections are not tagged with candidate entities, our IE method has to first identify candidate entities in the documents before classifying them.

The performance of our IE task is measured by three important metrics: *Precision*, *Recall* and F_1 *measure*. Let n_{tp} , n_{fp} , and n_{fn} be the number of entities correctly extracted, number of entities wrongly extracted, and number of entities missed respectively. *Precision*, *recall* and F_1 *measure* are defined as follows:

$$Precision = \frac{n_{tp}}{n_{tp} + n_{fp}}$$

$$Recall = \frac{n_{tp}}{n_{tp} + n_{fn}}$$

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

4 Proposed Method

4.1 Overview

Like other IE methods, we divide our proposed IE method into two steps: the *learning step* and the *extraction step*. The former learns the extraction model for the target entities in the desired semantic category using the training documents and their target entities. The latter applies the learnt extraction model on other documents and extract new target entities.

The learning step consists of the following smaller steps.

1. *Document parsing*: As the target entities are perpetrators, they usually appear as noun-phrases in the documents. We therefore parse all the sentences in the document. To break up a document into sentences, we use the SATZ software [10]. As a noun-phrase could be nested within another noun-phrase in the parse tree, we only select all the simple noun-phrases as candidate entities. The candidate entities from the training documents are further grouped as positive entities if their corresponding noun-phrases match the perpetrator answer keys. The rest are used as negative entities.

2. *Feature acquisition*: This step refers to deriving features for the training target entities, i.e., the noun-phrases. We will elaborate this step in Section 4.2.
3. *Extraction model construction*: This step refers to constructing the extraction model using some machine learning technique. In this paper, we explore the use of SVM to construct the extraction model (or classification model).

The classification step performs extraction using the learnt extraction model following the steps below:

1. *Document parsing*: The sentences in every test document are parsed and simple noun phrases in the parse trees are used as candidate entities.
2. *Feature acquisition*: This step is similar to that in the learning step.
3. *Classification*: This step applies the SVM classifier to extract the candidate entities.

By identifying all the noun-phrases and classifying them into positive entities or negative entities, we transform the IE problem into classification problem. To keep our method simple, we do not use co-referencing to identify pronouns that refers to the positive or negative entities.

4.2 Feature Acquisition

We acquire for each candidate entity the features required for constructing the extraction model and for classification. To ensure that the extraction model will be able to distinguish entities belonging to a semantic category or not, it is necessary to acquire a wide spectrum of features. Unlike the earlier work that focus on features that are mainly derived from within the entities [1] or the linear sequence of words surrounding the entities [4], our method derives features from syntactic structures of sentences in which the candidate entities are found.

We divide the entity features into two categories:

- **Content features**: These refer to the features derived from the candidate entities themselves. At present, we only consider terms appearing in the candidate entities. Given an entity $e = w_1w_2 \cdots w_n$, we assign the content feature $f_i(w) = 1$ if word w is found in e .
- **Context features**: These features are obtained by first parsing the sentences containing a candidate entity. Each context feature is defined by a fragment of syntactic structure in which the entity is found and words associated with the fragment.

In the following, we elaborate the way our context features are obtained. We first use the CMU’s Link Grammar Parser to parse a sentence [14]. The parser generates a parse tree such as the one shown in Figure 2.

A parse tree represents the syntactic structure of a given sentence. Its leaf nodes are the word tokens of the sentence and internal nodes represents the syntactic constituents of the sentence. The possible syntactic constituents are S (clause), VP (verb phrase), NP (noun phrase), PP (prepositional phrase), etc..

```

(S (NP Two terrorists)
  (VP (VP destroyed
      (NP several power poles)
      (PP on
        (NP 29th street)))
    and
    (VP machinegunned
      (NP several transformers)))
.)

```

Fig. 2. Parse Tree Example

For each candidate entity, we can derive its context features as a vector of term weights for the terms that appear in the sentences containing the noun-phrase. Given a sentence parse tree, the weight of a term is assigned as follows. Terms appearing in the sibling nodes are assigned the weights of 1.0. Terms appearing in the higher level or lower level of the parse tree will be assigned smaller weights as they are further away from the candidate entity. The feature weights are reduced by half for every level further away from the candidate entity in our experiments. The 50% reduction factor has been chosen arbitrarily in our experiments. A careful study needs to be further conducted to determine the optimal reduction factor. For example, the context features of the candidate entity “several power poles” are derived as follows⁵.

Table 1. Context features and feature weights for “several power poles”

Label	Terms	Weight
PP	on	1.00
NP	29th street	0.50
VP	destroyed	0.50
NP	Two terrorists	0.25

To ensure that the included context features are closely related to the candidate entity, we do not consider terms found in the sibling nodes (and their subtrees) of the ancestor(s) of the entity. Intuitively, these terms are not syntactically very related to the candidate entity and are therefore excluded. For example, for the candidate entity “several power poles”, the terms in the subtree “and machinegunned several transformers” are excluded from the context feature set.

⁵ More precisely, stopwords removal and stemming are performed on the terms. Some of them will be discarded during this process.

If an entity appears in multiple sentences in the same document, and the same term is included as context features from different parse trees, we will combine the context features into one and assign it the highest weight among the original weights. This is necessary to keep one unique weight for each term.

4.3 Extraction Model Construction

To construct an extraction model, we require both positive training data and negative training data. While the positive training entities are available from the answer key file, the negative training entities can be obtained from the noun phrases that do not contain any target entities. Since pronouns such as “he”, “she”, “they”, etc. may possibly be co-referenced with some target entities, we do not use them as positive nor negative training entities. From the training set, we also obtain a *entity filter dictionary* that consists of noun-phrases that cannot be perpetrators. These are non-target noun-phrases that appear more than five times in the training set, e.g., “dictionary”, “desk” and “tree”. With this filter, the number of negative entities is reduced dramatically. If a larger number is used, fewer noun-phrases will be filtered causing a degradation of precision. On the other hand, a smaller number may increase the risk of getting a lower recall.

Once an extraction model is constructed, it can perform extraction on a given document by classifying candidate entities in the document into perpetrator or non-perpetrator category. In the extraction step, a candidate entity is classified as perpetrator when the SVM classifier returns a positive score value.

5 Experiments and Results

5.1 Datasets

We used MUC-4 dataset in our experiments. Three files (muc34dev, muc34tst1 and muc34tst2) were used as training set and the remaining two files (muc34tst3 and muc34tst4) were used as test set. There are totally 1500 news documents in the training set and 100 documents each for the two test files. For each news document, there are zero, one or two perpetrators defined in the answer key file. Therefore, most of the noun phrases are negative candidate entities.

To avoid severely unbalanced training examples, we only considered the training documents that have at least one perpetrator defined in the answer key files. There are 466 training documents containing some perpetrators. We used all the 100 news documents in the test set since the classifier should not know if a test document contains a perpetrator. The number of documents used, number of positive and negative entities for the training and test sets are listed in Table 2. From the table, we observe that negative entities contribute about 90% of the entities of training set, and around 95% of the test set.

5.2 Results

We used *SVM^{light}* as our classifiers in our experiment [6]. The *SVM^{light}* is an implementation of Support Vector Machines (SVMs) in C and has been widely

Table 2. Documents, positive/negative entities in training/test data set

Dataset	Documents	Positive Entities	Negative Entities
Train	466	1003	9435
Tst3	100	117	2336
Tst4	100	77	1943

used in text classification and web classification research. Due to the unbalanced training examples, we set the *cost-factor* (parameter j) of SVM^{light} to be the ratio of number of negative entities over the number of positive ones. The *cost-factor* denotes the proportion of cost allocated to training errors on positive entities against errors on negative entities. We used the polynomial kernel function instead of the default linear kernel function. We also set our threshold to be 0.0 as suggested. The results are reported in Table 3.

Table 3. Results on training and test dataset

Dataset	<i>Precision</i>	<i>Recall</i>	F_1 <i>measure</i>
Train	0.7752	0.9661	0.8602
Tst3	0.3054	0.4359	0.3592
Tst4	0.2360	0.5455	0.3295

As shown in the table, the SVM classifier performed very well for the training data. It achieved both high precision and recall values. Nevertheless, the classifier did not perform equally well for the two test data sets. About 43% and 54% of the target entities have been extracted for Tst3 and Tst4 respectively. The results also indicated that many other non-target entities were also extracted causing the low precision values. The overall F_1 measures are 0.36 and 0.33 for Tst3 and Tst4 respectively.

The above results, compared to the known results given in [11] are reasonable as the latter also showed not more than 30% precision values for both AutoSlog and AutoSlog-TS⁶. [11] reported F_1 measures of 0.38 which is not very different from ours.

The rather low F_1 measures suggest that this IE problem is quite a difficult one. We, nevertheless, are quite optimistic about our preliminary results as they clearly show that the IE problem can be handled as a classification problem.

⁶ The comparison cannot be taken in absolute terms since [11] used a slightly different experimental setup for the MUC-4 dataset.

6 Conclusions

In this paper, we attempt to extract perpetrator entities from a collection of untagged news documents in the terrorism domain. We propose a classification-based method to handle the IE problem. The method segments each document into sentences, parses the latter into parse trees, and derives features for the entities within the documents. The features of each entity are derived from both its content and context. Based on SVM classifiers, our method was applied to the MUC-4 data set. Our experimental results showed that the method performs at a level comparable to some well known published results.

As part of our future work, we would like to continue our preliminary work and explore additional features in training the SVM classifiers. Since the number of training entities is usually small in real applications, we will also try to extend our classification-based method to handle IE problems with small number of seed training entities.

References

1. S. Baluja, V. Mittal, and R. Sukthankar. Applying machine learning for high performance named-entity extraction. *Computational Intelligence*, 16(4):586–595, November 2000.
2. S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, pages 148–155, Bethesda, Maryland, November 1998.
3. D. Freitag. Information extraction from HTML: Application of a general machine learning approach. In *Proceedings of the 15th Conference on Artificial Intelligence (AAAI-98) 10th Conference on Innovation Applications of Artificial Intelligence (IAAI-98)*, pages 517–523, Madison, Wisconsin, July 1998.
4. D. Freitag and A. K. McCallum. Information extraction with hmms and shrinkage. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 31–36, Orlando, FL., July 1999.
5. T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998.
6. T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.
7. J.-T. Kim and D. I. Moldovan. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Transaction on Knowledge and Data Engineering*, 7(5):713–724, 1995.
8. MUC. Proceedings of the 4th message understanding conference (muc-4), 1992.
9. I. Muslea. Extraction patterns for information extraction tasks: A survey. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 1–6, Orlando, Florida, July 1999.
10. D. D. Palmer and M. A. Hearst. Adaptive sentence boundary disambiguation. In *Proceedings of the 4th Conference on Applied Natural Language Processing*, pages 78–83, Stuttgart, Germany, October 1994.

11. E. Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 1044–1049, Portland, Oregon, 1996.
12. E. Riloff. An empirical study of automated dictionary construction for information extraction in three domains. *Artificial Intelligence*, 85(1-2):101–134, 1996.
13. E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level boot-strapping. In *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 1044–1049, 1999.
14. D. Sleator and D. Temperley. Parsing english with a link grammar. Technical Report CMU-CS-91-196, Computer Science, Carnegie Mellon University, October 1991.
15. S. Soderland. Learning information extraction rules for semi-structured and free text. *Journal of Machine Learning*, 34(1-3):233–272, 1999.
16. V. N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, Heidelberg, DE, 1995.