Named Entity Recognition using Support Vector Machines

Silviu Cucerzan
Department of Computer Science
Johns Hopkins University
silviu@cs.jhu.edu

Abstract

Identifying and classifying personal, geographic, institutional or other names in a text is an important task for numerous Natural Language Processing applications. This paper describes the implementation of Support Vector Machines into a language-independent bootstrapping algorithm which is based on iterative learning and re-estimation of contextual and morphological patterns captured in hierarchically smoothed trie models.

1 Introduction

The ability to determine the named entities in a text has been established as an important task for several natural language processing areas, including information retrieval, machine translation, information extraction and language understanding. A separate named entity recognition task was developed for the 1995 Message Understanding Conference (MUC-6) and the best systems achieved impressive accuracy, F-measure approaching 95%, which is close to the human performance on this task: 96% (Sundheim 1995). What should be underlined here is that these systems were trained for a specific domain and a particular language (English), typically making use of hand-coded rules, taggers, parsers and semantic lexicons. Even the systems that do not make use of extensive knowledge about a particular language, such as Nominator (Choi et al., 1997), still use large lists of names, exceptions, personal and organizational identifiers.

Our aim has been to build a maximally language-independent system for both named-entity identification and classification, using minimal information about the source language. The applicability of AI-style algorithms and supervised methods is limited in the multilingual case because of the cost of knowledge databases and manually annotated corpora. Therefore, a much more suitable approach is to consider a bootstrapping algorithm. In terms of world knowledge, the simplest and most relevant resource for this task is a database of known names. For each entity class to be recognized and tagged, it is assumed that the user can provide a short list (on the order of a few hundreds) of relatively unambiguous examples (seeds).

2 The Model

We present in this paper the implementation of Support Vector Machines into a language-independent bootstrapping algorithm that starts form lists of entity examples and a large unannotated corpus. The core algorithm used is the one presented in (Cucerzan and Yarowsky 1999) with a few modifications. Therefore, most of this presentation will focus on these modifications rather than on the main algorithm. The proposed modifications comprise: a new smoothing formula for computing the distributions along the paths in the tries, a new 'soft' discourse segmentation method and an SVM-based algorithm for entity classification.

2.1 Word-Internal and Contextual Information

The proposed method relies on both *word internal* and *contextual* clues as relatively independent evidence sources that drive the bootstrapping algorithm. The first category refers to the morphological structure of the word and makes use of the paradigm that for certain classes of entities some prefixes and suffixes are good indicators. Such morphological information is automatically learned during bootstrapping.

Contextual patterns (e.g. "*Mr.*", "*in*" and "*mayor of*" in left context) are also clearly crucial to named entity identification and classification, especially for names that do not follow a typical morphological pattern for their word class, are of foreign origin, or polysemous (for example "*Washington*").

2.2 Discourse Segmentation. One Sense per Discourse

Clearly, in many cases, the context for only one occurrence of a new word and its morphological information is not enough to make a decision. But, as noted in Katz (1996), a newly introduced entity will be repeated, "if not for breaking the monotonous effect of pronoun use, then for emphasis and clarity". Moreover, the number of instances of a new entity is not necessarily associated with the document length but with the importance of the entity with regard to the subject/discourse. We would

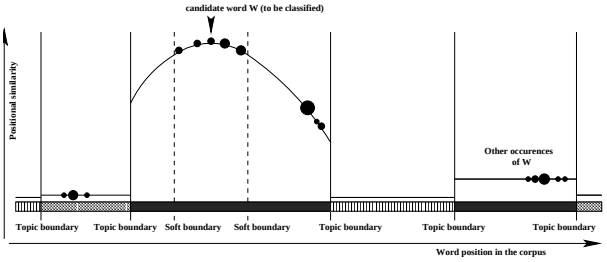

Figure 1: Using contextual clues from all instances of a word in a corpus (each instance is pictured as a ball with the diameter representing the confidence of the classification of that instance using only local contextual information)

like to use this property in conjunction with the *one sense per discourse* tendency noted by Gale, Church and Yarowsky (1992), who showed that words strongly tend to exhibit only one sense in a document/discourse, i.e. by gathering contextual information about the entity from each of its occurrences in the text and using morphological clues as well, we expect to classify entities more effectively than if they are considered in isolation, especially those that are very important with regard to the subject. Unfortunately, the latter paradigm can not be applied when analyzing large corpora that do not have document boundaries, like the Hansards (for example, word "*Emerson*" appears as surname, city, and company name in this corpus) and a segmentation algorithm should be considered, so that all the instances of a name in a segment have a high probability of belonging to the same class. Our approach is to consider a 'soft' segmentation, which is word-dependent and does not compute topic/document boundaries but regions for which the contextual information for all instances of a word can be used jointly when making a decision. This is viewed as an alternative to the classical topic/document segmentation approach and can be used in conjunction with a language-independent segmentation system (see Fig. 1) like the one presented by Amithay et al. (1997).

The probability of an entity class c for a candidate word w at position pos_i is computed considering all instances of that word (positions $pos_1, ..., pos_n$) in the corpus using the classification confidence of each instance (Z is a normalization factor):

$$P(c|w, pos_i) = \frac{1}{Z} \sum_{j=1...n} P_{local}(c|w, pos_j) \cdot sim(pos_i, pos_j) \cdot confidence(pos_j)$$

where the positional similarity sim encodes physical distance and topic.

2.3 Tokenized Text vs. Plain Text

There are two basic alternatives for handling a text. The first one is to tokenize it and classify the individual tokens or group of tokens. This alternative works for languages that use word separators (such as spaces or punctuation), where a relatively simple set of separator patterns can adequately tokenize the text. The second alternative, used in this research, is to classify entities simply with respect to a given starting and ending character position, without knowing precisely the word boundaries, but just the probability (that can be learned automatically) of a boundary given the neighboring contexts. This second alternative works for all languages including Chinese, where no separators between the words are typically used. Another advantage of this method is that single and multi-word entities can be handled in the same way. Although we do not consider an explicit tokenization, we will refer to a start and end point bounded portion of text being analyzed (in order to determine if it represents a named entity or not) as a token, for the simplicity of the presentation.

2.4 Trie Structures for Both Morphological and Contextual Information

Character-based tries provide an effective, efficient and flexible data structure for storing both contextual and morphological patterns and statistics. They are very compact representations and support a natural hierarchical smoothing procedure for distributional class statistics. In our implementation, each ramification or terminal node contains a probability distribution which encodes the probability of entity classes given the sistring corresponding to the path from the root to that node. Each distribution also has two standard classes, named "questionable" (unassigned probability mass in terms of entity classes, to be motivated below) and "non-entity" (common words). More details about this representation can be found in (Cucerzan and Yarowsky 1999).

Two tries are used for internal morphological representation of tokens, in prefix and suffix form. Other two are used for the left and right context of tokens. For right context, the letters are introduced in the trie in normal order (from left to right in the case of Indo-European languages), for left context they are considered in the reversed order (see the example in fig. 2).

The tries are linked together into two bipartite structures, morphological prefix with left context and morphological suffix with right context, by attaching to each node a list of links to the tokens/contexts with/in which the sistring corresponding to that node has been seen in the text.

For reasons that will be explained later, raw counts are kept for the distributions.

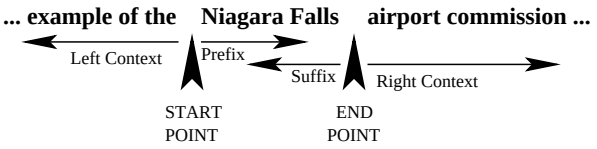

Figure 2: The way tokens and contexts are introduced in the four tries (arrows indicate the direction letters are considered)

2.5 The Bootstrapping Algorithm

The basic concept of the bootstrapping procedure is to iteratively leverage relatively independent sources of information. Beginning with some seed names for each class, the algorithm learns contextual patterns that are indicative for those classes and then iteratively learns new class members and word-internal morphological clues. Through this cycle, probability distributions for class given token, prefix/suffix or context are incrementally refined. More details are given in (Cucerzan and Yarowsky 1999).

2.6 Unassigned Probability Mass as Opposed to the Classical Maximum Entropy Principle

When faced with a highly skewed observed class distribution for which there is little confidence due to small sample size, a typical response to this uncertainty in statistical machine learning systems is to back-off or smooth to the more general class distribution, which is more uniform. Unfortunately, this representation is difficult to distinguish from a conditional distribution based on a very large sample (and hence estimated with confidence) that just happens to have a similar fairly uniform true distribution. One would like a representation that does not obscure this distinction, and represents the uncertainty of the distribution separately.

We resolve this problem while retaining a single probability distribution over classes by adding a separate "questionable" (or unassigned) cell that reflects the uncertainty of the distribution. Probability mass continues to be distributed among the remaining class cells proportional to the observed distribution in the data, but with a total sum (≤ 1) that reflects the confidence in the distribution and is equal to $1 - P_{node}(questionable)$.

By explicitly representing the uncertainty in a given class distribution, incremental learning essentially becomes the process of gradually shifting probability mass from questionable to one of the primary categories. As an important consequence, the whole bootstrapping procedure can be done in only one iteration over the known seeds.

2.7 Smoothing

The probability of a token/context as being in or indicating a class is computed along the path from the root to the terminal node corresponding to that token/context. In this way, effective smoothing can be realized for rare tokens or contexts. A new smoothing formula taking advantage of the distributional representation of uncertainty is presented below.

Considering a token/context formed from characters $l_1 l_2 ... l_n$, (i.e. the path in the trie is $root - l_1 - l_2 - ... - l_n$) the general smoothing model is given by the recursive formula:

$F(class_j|l_1 l_2 ... l_i) = f(class_j|l_1 l_2 ... l_i) +$
$\beta P_{node}(questionable|l_1 l_2 ... l_i)^{\alpha} F(class_j|l_1 l_2 ... l_{i-1})$,

where $\beta \in [0,1]$ and $\alpha \geq 1$ are language-dependent parameters.

The symbol F is used instead of P since we have raw distributions (frequencies) and a normalization step is needed to compute the final probability distribution.

2.8 The Main Algorithm

Briefly, the main algorithm can be divided into five stages, which are summarized below.

Stage 0: build the initial training list of class representatives (performed only once for each language/task)

Stage 1: extract the morphological and context tries from a large unannotated corpus

Stage 2: introduce the training information in the tries and re-estimate the distributions by bootstrapping

Stage 3: identify and classify the named entities in the text using competing classifiers

Stage 4: update the entity and context training space, using the new extracted information

3 Support Vector Machines

As presented in the preceding paragraph, the text is re-analyzed sequentially in stage 3 of the algorithm, for each candidate token a classification decision being made. The information used for this decision is represented by the two morphological distributions corresponding to the token and the contextual distributions of all instances of the respective token in the corpus. Using the interpolation formula presented in paragraph 2.2, we obtain two contextual distributions. Also, we keep the best (in terms of questionable mass) distributions in the 'soft bounded' window (see Fig. 1) corresponding to the given token for left and right context.

Therefore, a vector composed of 6 distributions over entity classes is available for the classification purpose for each candidate token. The decision with regard to the presence of an entity and its classification is made based on this distributional information.

We have replaced the voting scheme presented in (Cucerzan and Yarowsky 1999) with an SVM approach, by training a number of classifiers equal to the number of entity classes, each of them making

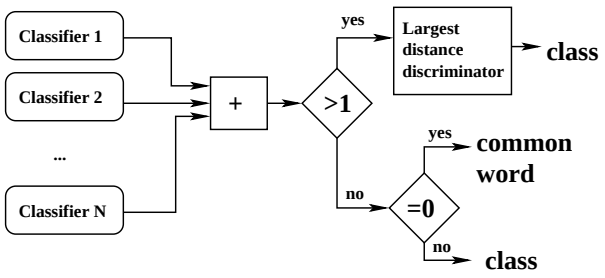

Figure 3: Combining the binary SV classifiers

a binary decision about a word being (+1) or not being (0) in a particular class. These classifiers are then combined into one 'largest distance' classifier as described in (Burges 1998), with the following twist: if only one of the binary classifiers outputs a positive result then the token is considered to be an entity in the class corresponding to that classifier (see Fig. 3). This modification is motivated by the fact that positive means an entity class, while null represents all other entity classes and common words; therefore, comparing the distances to the separating surfaces for a positive and a null output is not really meaningful.

There is a major problem that has to be solved in order to embed SVMs in the presented algorithm: the initial training data is represented by seed-word lists, i.e. lists of exemplars for each entity, rather then labeled vectorial distributions necessary to train the SVMs. This implies that the initial input data has to be transformed into usable vectorial representations. Our approach for this transformation is presented bellow:

Consider T the set of seed words available, S the set unambiguous seeds and $S' \subset S$ a small subset (of size about 10-20% of S). Perform stages 0 through 2 of the main algorithm using as seeds only $T - S'$. Output in stage 3 the distributions obtained for each instance of tokens from S' in the corpus and label these distributions with 1/0 according to the initial label for each of the binary classifiers.

Perform again stages 0 through 2 of the main algorithm using now the whole set of seeds T. Output in stage 3 the distributions obtained for each instance of tokens from S' in the corpus and label these distributions with 1/0 according to the initial label for each of the binary classifiers.

Use together the labeled distributions obtained in the two runs of the algorithm as training data for each binary classifier.

One of the problems we encounter with this approach is that for each classifier the number of training examples in the positive class (one particular entity class) is much smaller than the number of examples in the null class (all other classes and the common words). A way of overcoming this problem is to randomly select only as many null examples as positive examples.

4 Results and Future Work

The basic measures for evaluation of this work are precision and recall. Precision (P) represents the percentage of the entities that the system recognized which are actually correct. Recall (R) represents the percentage of the correct named entities in the text that the system identified. Both measures are incorporated in the F-measure, $F = 2PR/(P + R)$. It is important to observe that the accuracy of the SVM classifier is not identical to the standard F-measure, because classifying common words as non-entities is not actually rewarded, only misclassification of these words are punished.

We are not able to present the F-measure results for the new system at this moment, but only the accuracy results obtained by the SVMs on training and test sets.

We considered English as language and three entity classes: first name, last name, and place. The English side of the Hansards (14 million words) was used as unannotated corpus and 750 entities and common words constituted the seed list.

We chose polynomial kernel functions $K(x_i, x_j) = (x_i \cdot x_j + b)^d$ for their robusness and simplicity. The results presented in the following tables were obtained for the following choice of parameters: $d = 5$ (degree), $b = 0.1$ (offset), $C = 10$ (C-value).

We used the SvmFu software package developed by Rifkin (2001) because the of the availability of the code and training/testing and parameter setting options.

To obtain the training set of vectors, S' was considered to be composed of all seed words starting with letter A, B, or C. There were 9329 instances of these words in the unannotated corpus, as shown bellow:

L Name	F Name	Place	Non-entity	Total
952	722	3562	4093	9329

The test data was composed of all unambiguous seed words starting with letter D, E, F, or G. There were ... instances of these words in the corpus, distributed as shown below:

L Name	F Name	Place	Non-entity	Total
939	1107	1513	3138	6697

Three classifiers were trained using the labeled distributions that were output for the A-C words (one for each of the entity classes considered). The characteristics and performance of these classifiers on the training set are shown in the following table:

Classifier	L Name	F Name	Place
Number of support vectors	602	404	947
Correctly classified vectors	9113	9206	9141
Accuracy (%)	97.6	98.6	97.9

The performace of the classifiers on the held-out data set D-G is shown bellow:

Classifier	L Name	F Name	Place
Correctly classified vectors	6383	6310	6424
Accuracy (%)	95.3	94.2	95.9

These empirical results prove that SVMs can be succesfully used for the task of Named Entity classification. Work is in progress to modify the code of SvmFu (Rifkin), so that the SVMs trained as described in the previous paragraphs can be integrated in the fully-featured Named Entity software developed in 1999-2001 by Cucerzan and Yarowsky.

5 Conclusion

This paper has presented a minimally supervised learning method for named entity recognition, which uses an on-line bootstrapping procedure as the main algorithm and integrates Support Vector Machines as classifiers. The method makes use of hierarchically smoothed trie structures for modeling morphological and contextual probabilities effectively in a language independent framework, overcoming the need for fixed token boundaries or history lengths. The SVM approach proved successful in the early stages of this research and we are encouraged to integrate SVMs to the full extent in the proposed Named Entity Recognition system.

References

Amithay, E., K. Richmond, and A. Smith. 1997. Detecting Subject Boundaries within Text: A Language Independent Statistical ApproachIn *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 47-54.

Burges, C.J.C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition. In *Data Mining and Knowledge Discovery* 2, pp. 121-167

Choi, M., Y. Ravin, and N. Wacholder. 1997. Disambiguation of Proper Names in Text. In *Proceeding of the Fifth Conference on Applied Natural Language Processing*, pp. 202-208.

Cucerzan, S., and D. Yarowsky. 1999. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Proceedings of EMNLP/VLC '99*, pp. 90-99.

Day, D., and D. Palmer. 1997. A Statistical Profile of the Named Entity Task. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 190-193.

Gale, W., K. Church, and D. Yarowsky. 1992. One Sense per Discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, pp. 233-237.

Katz, S.M. 1996. Distribution of Context Words and Phrases in Text and Language Modeling. *Natural Language Engineering* 2(1):15-59.

Lewis, D. and W. Gale. 1994. A Sequential Algorithm for Training Text Classifiers. In *Proceedings of SIGIR'94*, pp. 3-12, Dublin.

Rifkin, R. 2001. SvmFu package, Version 2.004. *http://five-percent-nation.mit.edu/PersonalPages/rif/SvmFu/index.html*

Smola, A.J. and B.S. Schölkopf. 1998. A Tutorial on Support Vector Regression. In *NeuroCOLT2 Technical Report Series.*

Sundheim, B.M. 1995. Overview of Results of the MUC6 Evaluation. *Proceedings of the Sixth Message Understanding Conference*, pp. 13-31.

Yarowsky, D. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189-196.